MINISTERUL EDUCAȚIEI

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

mCLINIChelp-SISTEM SUPORT_PENTRU O CLINICĂ MEDICALĂ

LUCRARE DE LICENŢĂ

Absolvent: **Cristian POP**

Coordonator ştiinţific: **Asis. Ing. Cosmina IVAN**

2021

Cuprins

Capitolul 1. Introducere – Contextul proiectului

1.1. Motivație

Trăim într-o perioadă în care inovațiile tehnologice ne-au făcut să putem comunica între noi oricând si oriunde ne-am afla.Putem interacționa în diferite moduri datorită aplicațiilor de comunicare și de asemenea putem fi la curent cu tot ce se întamplă în orice colț al lumii. Această accelerare tehnologică ne-a adus multe beneficii și ne-a ușurat mult viața de zi cu zi, însă în anumite domenii, inclusiv în cel medical din țara noastră, evoluția tehnologiei este mai lentă, iar lucrurile se mișcă la fel de greu din lipsa unor aplicații inovatoare.

În sistemul medical lucrurile se dezvoltă mai lent, deși asistăm în ultima perioadă la o serie de îmbunătățiri în ceea ce privește digitalizarea sistemului medical din țara noastră, sunt binevenite multe soluții capabile să completeze paleta de aplicații mobile din acest domeniu, iar situația din ziua de azi nu face decât să îngreuneze și mai mult un sistem copleșit de abundența mare de cereri de spitalizare. Deși în cazurile unei urgențe, serviciile medicale ajung destul de repede, în cazul unor situații medicale care nu sunt urgențe, dar care ar necesita un control la doctor, timpul de așteptare este destul de mare. Astfel această aplicație are rolul de a conecta doctorii cu pacienții fără a mai fi nevoie ca pacienții să se deplaseze la spital pentru orice fel de problemă și să aștepte ore în șir pe coridoare.

În această perioadă dificilă pentru toată lumea provocată de pandemia de Sars-Cov2, s-a observat cât de importantă este comunicarea la distanță și rezolvarea problemelor medicale care nu sunt foarte grave și care nu necesită o deplasare de urgență la spital, dar care ar necesita sfaturile unui specialist, ar putea fi ușor rezolvate prin convorbirea cu un medic, prin intermediul sistemului de chat. Toate măsurile impuse de această pandemie au îngreunat foarte mult tratarea anumitor probleme medicale din cauza prioritizării acestui virus. O soluție la această problemă este e-health-ul, adică servicii și informații de sănătate transmise cu ajutorul internetului și a dispozitivelor care au conexiune la internet.Astfel datorită e-health-ului asistența medicală a devenit mai eficientă permițând pacienților și doctorilor să concretizeze ceea ce în trecut era imposibil de realizat. Dispozitivele mobile au ajuns să fie cele mai folosite gadgeturi de către oameni, astfel ele devenind un obiect important pentru digitalizarea sistemului medical pentru că acestea sunt la îndemâna majorității persoanelor în cea mai mare parte a zilei.

În concluzie,se urmărește crearea unei aplicații care să rezolve anumite probleme medicale care nu necesită neapărat o consultație fizică la doctor și care să poată fi utilizată de o clinică medicală, astfel s-ar putea reduce din numărul persoanelor care se deplasează până la clinică pentru anumite probleme medicale care nu sunt prioritare, dar care ar necesita o consultație sau un sfat,acestea putând fi realizate și de la distanță datorită digitalizarii.

1.2. Conținutul lucrării

În acest subcapitol se prezintă o scurtă descriere a conținutului fiecărui capitol din această lucrare.

- *Capitolul 1*-**Introducere**-în acest capitol se realizează o prezentare generală a domeniului în care se află aplicația precum și câteva probleme din domeniu pe care aplicația și le propune să le rezolve
- *Capitolul 2*-**Obiectivele proiectului-**se prezintă tema propriu-zisă a acestei lucrări precum și obiectivele principale pe care acest proiect le propune să le rezolve, dar si obiectivele secundare.
- *Capitolul 3*-**Studiu bibliografic-**se prezintă invovațiile din domeniul medical, conceptele de e-health și m-health, și de asemenea se prezintă aplicații similare precum și o analiză comparativă a acestor aplicații cu sistemul implementat.
- *Capitolul 4*-**Analiză si fundamentare teoretică-**sunt furnizate informații despre cerințele sistemului precum și despre tehnologiile care au fost folosite pentru realizarea sistemului. Mai sunt prezentate și cazurile de utilizare pentru fiecare tip de utilizator al sistemului
- *Capitolul 5*-**Proiectare de detaliu si implementare-**se prezintă modul în care s-a realizat implementarea fiecărei componente a sistemului, precum si diagrame mai importante cum ar fi cea de bază de date, clase, pachete.
- *Capitolul 6*-**Testare si validare-**conține informații despre testarea și validarea sistemului, dar si despre tehnologiile de testare utilizate
- *Capitolul 7*-**Manual de instalare si utilizare-**manual în care se detaliază resursele necesare pentru ca aplicația să poată fi instalată și rulată
- *Capitolul 8*-**Concluzii si dezvoltări ulterioare-**conține analiza rezultatelor obținute precum și dezvoltări ulterioare ale proiectului.

Capitolul 2. Obiectivele Proiectului

În acest capitol se prezintă obiectivele generale ale sistemului, precum și obiectivele secundare pe care sistemul și le propune să le implementeze.

2.1. Obiective generale

Scopul principal al proiectului este dezvoltarea unei aplicații medicale care să simplifice comunicarea dintre pacienți si doctori, precum și să rezolve anumite probleme medicale care nu sunt considerate urgențe și care se pot realiza și de la distanță cum ar fi un sistem de programări sau prescrierea de medicamente. Aplicația mai are rolul de a gestiona urgențele medicale folosind un sistem de notificări. Mai mult aplicația va fi disponibilă și unui administrator care se va ocupa cu gestionarea celorlalți utilizatori și anume pacienții, medicii și serviciul de ambulanță.

2.2. Obiective secundare

Ca si obiective secundare aplicația își propune să rezolve sau să îmbunătățească anumite neajunsuri din sistemul medical. Aceste soluții sunt grupate în funcție de tipul de utilizator și anume: pacient, doctor, serviciu de ambulanță.

Pentru pacient s-au propus următoarele obiective:

- **Programări**-De cele mai multe ori programarea la doctor înseamnă ca pacientul să sune la doctorul respectiv sau să se deplaseze până la cabinetul acestuia, iar acest lucru consumă mult timp, mai ales dacă pacientul nu reușește să îl găsească pe doctor rezultând în alte apeluri telefonice sau deplasări la cabinet. Metoda propusă este programarea online de unde pacientul va putea selecta doctorul la care dorește să își creeze programarea apoi data și ora. După ce a creat programarea va putea vizualiza toate programările.

- **Fișa medicală**-Fișa medicală este păstrată în acest moment în format fizic, fiind necesară pentru orice consultație la un doctor. Metoda propusă este ca fișa medicală să devină una electronică care să poată fi accesată foarte ușor de pe dispozitivul mobil de către fiecare pacient înregistrat. Astfel aceasta va fi tot timpul la îndemâna pacientului, reducând astfel timpul necesar căutării acesteia și de asemenea reducând și spațiul ocupat de aceasta.

- **Chat**-Comunicarea cu doctorul este foarte importantă în rezolvarea problemelor medicale. La fel ca la programări pentru a lua legătura cu un doctor este nevoie de un apel telefonic sau de o deplasare la cabinet chiar dacă problema pacientului nu este urgentă. Sistemul implementează un chat în care pacientul poate să comunice cu doctorul pentru a afla informații despre problema acestuia fără a se mai deplasa la cabinet.

Pacientul va putea trimite mesaje în orice moment al zilei, sistemul fiind întotdeauna disponibil.

- **Urgență medicală-**În anumite situații pacienții pot avea o problemă medicală și au nevoie de serviciul de ambulanță pentru a fi transportați la spital pentru îngrijiri medicale. Însă în anumite situații apelul către 112 poate fi ocupat sau timpul până ca pacientul să descrie locația acestuia(în cazul în care nu cunoaște zona în care se află) să fie foarte mare, iar acea durată poate fi decisivă în lupta dintre viață și moarte. Aplicația propune un sistem de notificări în care pacientul va comunica problema medicală a acestuia printr-o notificare către serviciul de ambulanță împreună cu datele sale de identificare,care se află în aplicație, și locația acestuia, reducând mult timpul necesar pentru a transmite informații.

- **Prescripție medicală-**De cele mai multe ori după o consultație medicii prescriu medicamente. Această prescripție care este scrisă pe hârtie poate fi foarte ușor pierdută. Sistemul își propune ca precripția de medicamente să fie digitală, pacientul având la dispoziție toate informațiile necesare despre medicamente cum ar fi numele, modul de administrare și efecte secundare.

Pentu doctor s-au propus următoarele obiective:

- **Chat-**Și pentru doctori va fi mai ușor să comunice cu pacienții și să le ofere informațiile de care au nevoie.

- **Programări-**Doctorii nu mai au nevoie de o agendă pentru a putea vedea programările, ci doar accesează aplicația unde apar toate programările pacienților.

- **Fișa medicală-**În cazul în care anumiți pacienți au avut numeroase antecedente medicale, fișa medicală a acestora conține mai multe file ca să poată cuprinde datele esențiale. Căutarea într-o astfel de fișă poate necesita timp și totodată aceste fișe medicale ocupă un spațiu semnificativ din biroul unui doctor. Pentru a reduce timpul și spațiul ocupat de fișa medicală sistemul își propune ca aceasta să fie digitală, astfel toate informațiile importante despre pacient sunt mult mai ușor de accesat.

- **Prescripție medicală-**În cazul în care anumiți doctori au foarte mulți pacienți, pentru aceștia este destul de dificil și necesită mult timp pentru a căuta la fiecare pacient fișa medicală cu scopul de a verifica anumite informații pentru a putea prescrie medicamentele corespunzătoare.

Sistemul propune ca prescrierea de medicamente să se realizeze digital, fiind mai ușor pentru doctori, dar și pentru pacienți.

Pentru serviciul de ambulanță s-au propus următoarele obiective:

- **Urgență medicală-**Serviciul de ambulanță trebuie anunțat de către dispecerat pentru a se deplasa la locația pacientului, iar în anumite cazuri lipsa informațiilor esențiale poate face diferența. Astfel sistemul își propune ca serviciul de ambulanță să primească o notificare de la pacient în care acesta specifică problema medicală și de asemenea această notificare să conțină și locația curentă a acestuia. Serviciul de ambulanță răspunde tot printr-o notificare către pacient cu durata de timp necesară pentru a ajunge la acesta, timpul de răspuns fiind foarte mic.

Capitolul 3. Studiu Bibliografic

În acest capitol se prezintă tehnologia în domeniul medical, conceptele de e-health si m-health, precum și câteva aplicații similare cu sistemul si o comparație între ele.

3.1. Tehnologia în domeniul medical

Evoluția tehnologiei de-a lungul timpului a avut un impact pozitiv în domeniul medicinei și a ajutat la accelerarea dezvoltării acesteia, oferind noi metode pentru tratarea pacienților sau pentru a putea detecta diferite afecțiuni medicale.

Rolul tehnologiei în acest domeniu este de a simplifica interacțiunea dintre doctori și pacienți, de a îi ajuta pe cei care au nevoie de îngrijiri medicale, personalul medical fiind înștiințat cu ajutorul dispozitivelor mobile folosind diferite aplicații, apeluri de urgență, SMS sau internetul. Un alt avantaj al tehnologiei reprezintă digitalizarea datelor, fapt care a transformat modul vechi de stocare a datelor pe hârtii, într-un mod nou și performant, datele fiind stocate și centralizate într-un sistem în care informația poate fi căutată și gasită mult mai ușor.

Un alt avantaj al dezvoltării tehnologiei în domeniul medical este reducerea costului prin automatizarea anumitor operații care în trecut trebuiau realizate de către oameni. De exemplu așa cum este specificat și în articolul [4] înainte să existe pompe de perfuzie asistentele medicale trebuiau să facă injecții la anumite intervale de timp. De când au apărut aceste pompe de perfuzie și au automatizat procesul timpul alocat asistentelor pentru aceste injecții a fost eliberat, ele având acel timp disponibil pentru alte activități. De asemenea după ce au fost produse pompele de perfuzie nu mai există alte costuri pentru programarea acestora. Astfel prețul scade, dar crește cota de piață si profitul astfel că producătorul poate să investească în alte tehnologii de producție și distribuție. Este de menționat faptul că aceste aparate sunt create pentru majoritatea pacienților, nu și pentru cazuri speciale cum ar fi o boală rară deoarece există costuri suplimentare în producția unui aparat care trebuie adaptat pentru anumite persoane.

Senzorii sunt un instrument cu o utilizare tot mai largă în domeniul medical datorită faptului că pot fi utilizați pentru monotorizarea pacienților de la distanță.Aceștia pot detecta semnale fizice, chimice si biologice, aceste semnale fiind trimise mai departe pentru a fi măsurate și memorate. Un exemplu de dispozitiv care conține senzori pentru memorat parametri de sănatate este smartwatch-ul.Acesta cât timp este purtat pe mână monitorizează pulsul proprietarului, numărul de pași efectuați în ziua respectivă, numărul de calorii arse, numărul de ore dormite. Acești parametri oferă utilizatorului o percepție despre modul său de viață și dacă acesta trebuie schimbat sau nu.

Inteligența artificială joacă și ea un rol important în dezvoltarea medicinei mai ales în domeniul procesării imaginilor.În trecut pentru fiecare radiografie medicul trebuia să o examineze și apoi să stabilească diagnosticul, iar acest lucru necesita timp, în prezent există algoritmi de învățare pentru clasificarea acestor imagini. Acești algoritmi se bazează pe un set extrem de mare de date de învățare oferind o precizie foarte mare în recunoașterea diferitelor boli. Astfel timpul pe care doctorii îl alocau pentru aceste diagnostice în funcție de radiografii s-a eliberat, ei având timp pentru alte activități medicale[5].

3.2. E-health

E-health este un termen care înglobează toate posibilitățile de digitalizare a serviciilor de sănătate. Printre aceste servicii se numară programări online, transmiterea de informații și fișiere de la distanță, monitorizarea pacienților de la distanță, stocarea datelor pentru a putea fi accesate mai ușor atât de către pacienți, cât și de personalul medical.

Acest concept este unul relativ nou pentru domeniul medicinei. Informația a fost întotdeauna o parte importantă în acest domeniu, stocarea datelor fiind necesară pentru a cunoaște starea de sănătate a unui pacient, precum și antecedentele medicale ale acestuia. Digitalizarea datelor este astfel posibilă datorită dezvoltării tehnologice, însă vine și cu provocări, iar conform articolului [1] acestea sunt capacitatea consumatorului de a interacționa cu un sistem online, posibilități îmbunătățite de transmisie a datelor între instituții și noi posibilități de comunicare între consumatori. De asemenea sporește oportunitatea de auto-îngrijire, astfel scade riscul și impactul bolilor și îmbunătățește cunoștințele utilizatorilor în domeniul medical[6].

După cum este prezentat și în articolul [1] principalele avantaje ale e-health-ului sunt:

- Eficiența-e-health-ul își propune să crească eficiența în domeniul medical și de asemenea să scadă costul.
- Îmbunătățirea calitații serviciilor-dacă crește eficiența atunci, nu doar că va fi redus costul, dar va fi si îmbunătățită calitatea serviciilor. Un exemplu de creștere a calității serviciilor este posibilitatea de a compara diferite servicii de sănătate de la diferiți furnizori, implicând astfel consumatorii ca putere suplimentară pentru asigurarea calitații, ei direcționând pacienții către serviciile cu cea mai bună calitate
- Educație-posibilitatea de a pune la dispoziție surse de informare pentru pacienți pentru educarea acestora și pentru a le oferi informații credibile despre tot ce înseamnă sănătate
- Încurajarea unei noi relații între pacienți și personalul medical unde deciziile sunt luate în mod comun
- Extinderea sferei de îngrijire a sănătății dincolo de granițele sale convenționale. E-health-ul permite consumatorilor acces la servicii de la furnizori globali. Aceste servicii variază de la sfaturi simple la informații despre produse farmaceutice.

3.3. M-health

M-health se referă la servicii de sănătate care pot fi obținute folosind dispozitivul mobil sau tableta.

Astăzi un dispozitiv pe care îl putem ține într-o singură mână și care deține o putere de calcul care, în urmă cu câteva decenii ar fi costat milioane de dolari și ar fi necesitat spațiul unei întregi camere pentru toate instrumentele necesare rulării acestuia. Dezvoltarea tehnologică a dispozitivelor mobile a făcut ca telefonul să devină un accesoriu indispensabil din viața oamenilor. Începând de la comunicarea cu alte persoane fiecare aspect al vieții noastre a fost transformat de aceaste dispozitive cum ar fi finanțe,

călătorii, divertisment, educație. De asemenea și domeniul medical este influențat de acestă dezvoltare prin faptul că apar pe piață tot mai multe aplicații care sunt create pentru îngrijirea sănătății noastre. Prin intermediul telefoanelor mobile, mai exact a smartphone-urilor care au capacitate și putere de calcul tot mai mare, monitorizarea, urmărirea și transmiterea valorilor de sănătate în timp real au oferit posiblitatea pentru diagnosticarea unor boli și gestionarea afecțiunilor cronice în afara unui cabinet medical sau a unui spital [3]. Astfel monitorizarea pacienților folosind dispozitivele mobile a devenit deja o opțiune viabilă. Aplicațiile destinate supravegherii sănătății ajută la colectarea datelor cumunicate sau să asiste persoanele cu dizabilități pentru a putea avea un trai independent. Dispozitivele mobile au fost, de asemenea utilizate pentru a urmări cu precizie ritmul cardiac și variabilitatea acestuia [7].

Potrivit ultimei statistici[1] din iunie 2021, 48,2% din populația lumii deține un smartphone, astfel că dispozitivele mobile joacă un rol important în domeniul medical. Există tot mai multe aplicații medicale care au rolul de a satisface nevoie utilizatorilor asupra serviciilor medicale de la distanță cum ar fi programările, urgențele medicale, gestionarea datelor, diagnostice, rezultatele analizelor, comunicarea mai rapidă și mai eficientă cu doctorii.

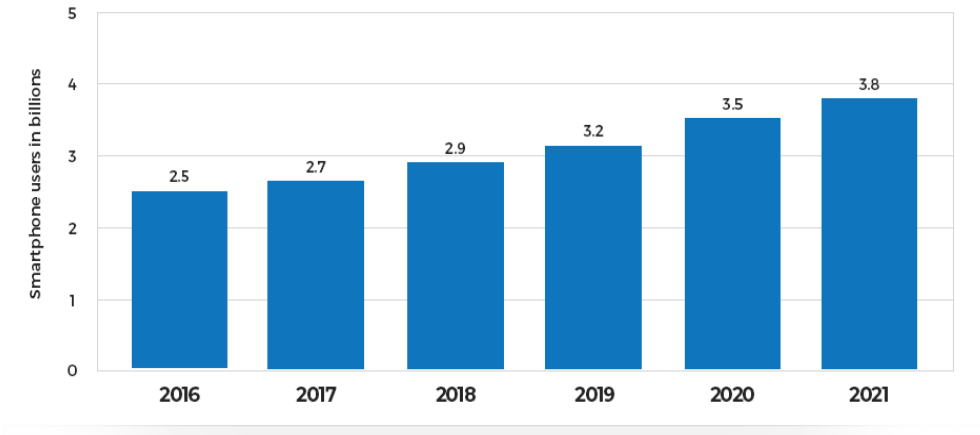


Fig 3. 1 Evoluția umărului de persoane care deține un smartphone[1]

Unul din cele mai importante aspecte legate de m-health este gestionarea urgențelor. Conform unei statistici[2], timpul mediu de răspuns al unei ambulanțe variază în funcție de gravitatea situației. Pentru urgențele din categoria 1, în care viața pacientului este pusă în joc, cum ar fi stop cardio-respirator timpul mediu de răspuns este de 7 minute, pentru categoria 2, care sunt clasificate ca fiind o urgență pentru o afecțiune gravă și care necesită o evaluare rapidă timpul mediu de răspuns este de 18 minute. Pentru cele din categoria 3, care sunt clasificate ca și urgențe, dar care nu pun viața pacientului în pericol, timpul de răspuns mediu este de 120 de minute, iar pentru ultima categorie, categoria 4, care nu sunt considerate urgențe, timpul de răspuns mediu este de 180 de minute. Luând în calcul și faptul că timpul petrecut la telefon în urma apelului către dispecerat variază între 1 și 5 minute, timpul de așteptare pentru primirea de îngrijiri medicale este destul de mare, iar acești timpi ar putea fi micșorați crescând astfel șansele la viață ale pacienților aflați în stare gravă. Pentru a micșora timpii se propune o

[1] https://www.bankmycell.com/blog/how-many-phones-are-in-the-world
[2] https://www.nuffieldtrust.org.uk/resource/ambulance-response-times

gestionare mai eficientă la telefon, iar lucrearea [2] prezintă o aplicație mobilă care atunci când utilizatorul are o urgență acesta trimite mesaj și apel către unul sau mai multe contacte de urgență, printre acestea poate fi și ambulanța și în care se transmit și datele personale ale acestuia care au fost introduse la înregistrarea în aplicație.

Mhealth-ul poate fi folosit și ca unealtă de învățare și dezvoltare. Conform unei estimări a OMS [8], există un deficit de 7,5 milioane de medici, farmaciști, asistente și alți angajați de specialitate în domeniul medical, iar această cifră va crește până la 12.9 milioane în următoarele decenii. Un factor important care trebuie luat în considerare este educația, mai exact calitatea, eficiența și sustenabilitatea acesteia. Din cauza calității educației în anumite universități , viitorii studenți optează pentru altă universitate din altă zonă, sau chiar alt domeniu decât cel medical, creând un deficit de specialiști în zona respectivă. Însă ultimele dezvoltări tehnologice oferă o soluție pentru această problemă prin utilizarea mHealth-ului ca o unealtă de învățare. Există o multitudine de aplicații destinate domeniului medical în care utilizatorii au acces la informații actuale, realizate de specialiști în domeniu menite să învețe noua generație bazele medicinei și să îi ajute să se specializeze pe un anumit domeniu.

3.4. Aplicații similare

În această secțiune se vor prezenta câteva aplicații care au fost identificate cu scopul realizării unei poziționări cât mai corecte a produsului în piață, dar și pentru a identifica funcționalități utile și abordări în implementarea lor.

3.4.1. mClinic

mClinic[3] este o aplicație disponibilă atât pe iOS, cât și pe Android și este destinată atât doctorilor, cât și pacienților. Doctorii trebuie să se autentifice pentru a putea folosi aplicația, însă pacienții pot folosi aplicația fără a avea nevoie de un cont.

Pacienții pot vedea specialitățile medicale, iar în funcție de acestea pot vedea doctorii disponibili. De asemenea folosind Google Maps aceștia pot vedea doctorii și spitalele din zona acestora. Mai mult aceștia pot crea o programare la un anumit doctor selectând ziua și ora la care doresc să își creeze programarea.

Doctorul poate vizualiza programările, le poate șterge și poate adăuga intervalul de timp pentru durata programării.

[3] http://www.mclinicapp.com/

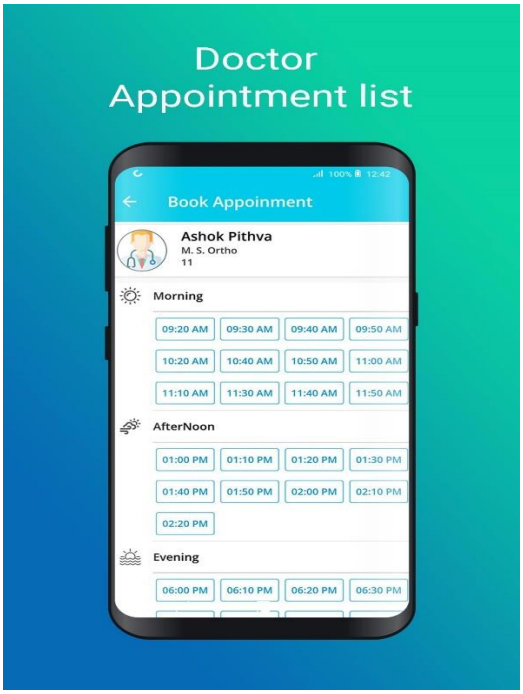


Fig 3. 2 mClinic

3.4.2. Doctor on Demand

Doctor On Demand[4] este o aplicație mobilă disponibilă 24/7 care oferă consultații online cu doctori de specialitate. Pentru o consultație trebuie ca utilizatorul să creeze o programare.

Aplicația dispune de creearea unei programări la un doctor de specialitate, de o hartă interactivă unde se pot vedea cabinete medicale și laboratorare din apropierea utilizatorului. Aplicația mai dispune și de un sistem de chat pentru o comunicare mai ușoară între doctor și pacient, un istoric medical al programărilor trecute și posibilitatea de a distribui fișiere între doctor și pacient.

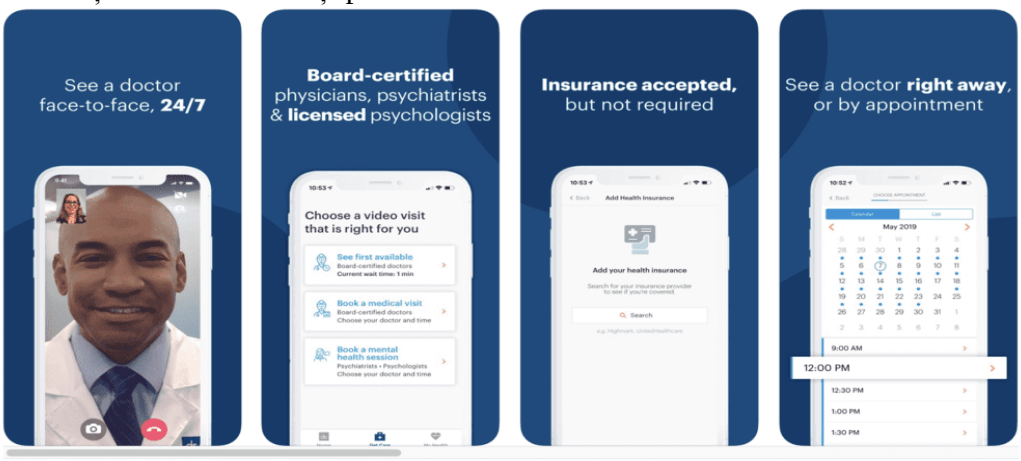


Fig 3. 3 Doctor on Demand

[4] https://www.doctorondemand.com/

3.4.3. MyChart

MyChart[5] este o aplicație care oferă informații despre sănătate și ajută la îngrijirea sănătății utilizatorului și a celor apropiați lui.

Aplicația permite utilizatorilor să comunice cu o echipă specializată în îngrijirea persoanelor bolnave, să acceseze rezultatele testelor efectuate, istoricul imunizărilor și alte informații medicale. De asemenea se pot vedea costurile îngrijirilor medicale, iar aplicația permite vizualizarea și plata serviciilor medicale.

Pentru a putea folosi această aplicație utilizatorul trebuie să își creeze un cont folosind asigurarea medicală.

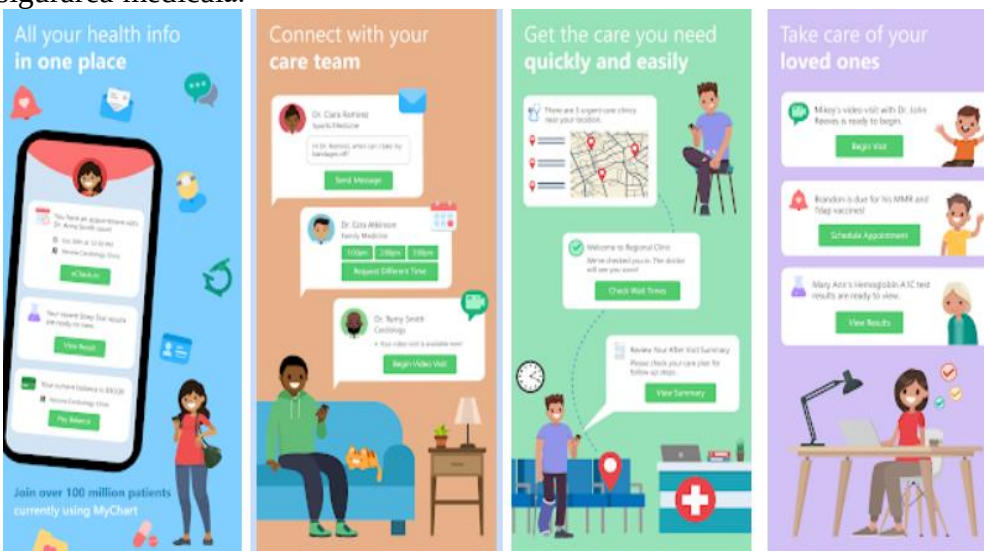


Fig 3. 4 MyChart

3.4.4. MedicalID

Medical ID[6] este o aplicație care permite crearea de profiluri medicale care sunt accesibile din ecranul de blocare. În caz de urgență profilurile permit accesul rapid la informații vitale, cum ar fi alergiile, grupa de sânge, contacte medicale care sunt esențiale atunci când medici, asistente sau medici din ambulanță trebuie să acționeze rapid. Mai mult se poate partaja locația cu contactele de urgență chiar și când aplicația este închisă.

[5] https://www.mychart.com/
[6] https://play.google.com/store/apps/details?id=app.medicalid.free&hl=en_US&gl=US

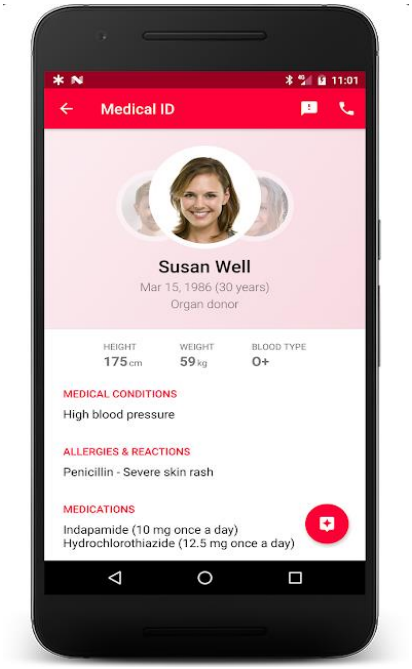


Fig 3. 5 MedicalID

3.4.5. Analiză comparativă

Tabelul de mai jos reprezintă o analiză comparativă între aplicațiile prezentate mai sus și aplicatia mCLINIChelp.

Tabel 3. 1 Comparație între sistemul propus si aplicații similare

Funcționalitate	mClinic	Doctor On Demand	My Chart	Medical ID	mCLINIChelp
Programare consultație	Da	Da	Nu	Nu	Da
Sistem de chat între doctor si pacient	Nu	Da	Da	Nu	Da
Localizare GPS	Da	Da	Da	Da	Da
Prescripție medicală	Nu	Nu	Da	Da	Da
Vizualizare fișă medicală	Nu	Da	Da	Da	Da
Sistem de notificări	Nu	Nu	Nu	Da	Da
Gestionare urgențe	Nu	Nu	Nu	Da	Da

Se pot observa diferențele dintre sistemul mHealth și sistemele prezentate

- Doar două sisteme oferă posibilitatea comunicării directe dintre doctor și pacient printr-un sistem de chat și anume Doctor On Demand și MyChart
- Doar un sistem oferă posibilitatea gestionării urgențelor medicale și un sistem de notificări și anume MedicalID
- Doar două sisteme pun accentul pe programare online și anume mClinic și Doctor On Demand

Capitolul 4. Analiză şi Fundamentare Teoretică

4.1. Cerințele sistemului

În acest capitol vor fi prezentate cerințele sistemului, cele funcționale și cele non-funcționale aferente sistemului.

4.1.1. Cerințe funcționale

Tabel 4. 1 Cerințele funcționale ale sistemului

Nr. Funcționalitate	Utilizator	Cerința
F1.	Administrator/doctor/pacient/serviciu ambulanță	**Autentificare**-utilizatorul se poate atutentifica folosind adresa de mail și parola existente în baza de date
F2.	Pacient	**Creare cont**-dacă nu are cont pacientul are posibilitatea de a-și crea cont introducând datele necesare: nume, prenume, data nașterii, adresa, telefon, mail, parolă. Dacă mailul este utilizat de către alt cont atunci se va primi un mesaj de eroare
F3.	Administrator	**Operatii CRUD utilizatori**-administratorul are posiblitatea de a vizualiza, adăuga, actualiza și șterge ceilalți utilizatori
F4.	Pacient/doctor	**Programare**-pacientul are posibilitatea de a crea o programare la un anumit doctor, de a vizualiza programările create. Doctorul poate vizualiza programările sale.
F5.	Pacient/doctor	**Chat**-pacienții pot comunica cu doctorii și invers folosind sistemul de chat
F6.	Pacient/doctor	**Fișa medicala**-pacienții își pot vizualiza fișa lor medicală; doctorii pot vizualiza fișele medicale ale pacienților și le pot edita
F7.	Pacient/doctor	**Prescripție**-pacienții pot vizualiza prescripțiile care trebuie administrate; doctorii pot prescrie pacienților medicamente, specificând efectele adverse,

		administrarea acetora și prețul
F8.	Pacient/serviciu ambulanță	**Urgență medicala-**pacienții pot solicita printr-o notificare serviciul de ambulanță, acesta le va răspunde tot printr-o notificare cu durata de timp necesară pentru a ajunge la aceștia

4.1.2. Cerințe non-funcționale

Tabel 4. 2 Cerințele non-funcționale ale sistemului

Nr. Funcționalitate	Cerință	Descriere
NonF1.	Securitate	Aceasta este probabil cea mai intuitivă cerință a sistemului deoarece datele din sistem trebuie protejate împotriva unor persoane neautorizate. Date precum parola trebuie încriptate în baza de date ,se va folosi tokenul oferit de FirebaseAuthentication pentru accesul în aplicație. Mai mult accesul utilizatorilor în aplicație va fi restricționat în funcție de tipul de utilizator.
NonF2.	Disponibilitate	Reprezintă timpul în care aplicația va fi disponibilă utilizatorilor. Aceasta va fi tot timpul disponibilă, cu condiția ca utilizatorul să aibă conexiune la internet.
NonF3.	Utilizabilitate	Indică gradul de usurință în utilizarea aplicației. Aceasta trebuie să fie ușor de folosit, să aibă o interfață grafică cu un design intuitiv ca fiecare utilizator să înțeleagă cum se realizează fiecare operație.
NonF4.	Accesibilitate	Aplicația se poate accesa foarte ușor folosind un dispozitiv mobil care rulează Android și are conexiune la internet. Pentru a accesa aplicația utilizatorul trebuie să se autentifice sau să își creeze cont(cazul pacientului), iar după va putea folosi aplicația.

NonF5.	Performanță	Indică eficiența sistemului prin următoarele caracteristici: timpul de răspuns al sistemului la diferite operații ale utilizatorului și numărul de utilizatori concurenți pe care sistemul îi poate suporta. Acest număr este de aproximativ un milion de utilizatori concurenți[7]
NonF6.	Extensibilitate	Aplicația permite adăugarea cu ușurință a unor dezvoltări ulterioare pe viitor.

4.2. Cazuri de utilizare

Cazurile de utilizare oferă o privire de ansamblu a funcționalităților care se doresc să fie oferite de sistem. Acestea arată cum interacționează sistemul cu unul sau cu mai mulți actori și astfel ne ajută să explicăm comportamentul sistemului.

4.2.1. Actori

În acest subcapitol sunt prezentați actorii sistemului și anume, administratorul, pacientul, doctorul si serviciul de ambulanță. Aceste diagrame prezintă toate acțiunile posibile pe care le pot avea acești utilizatori.

4.2.1.1. Administrator

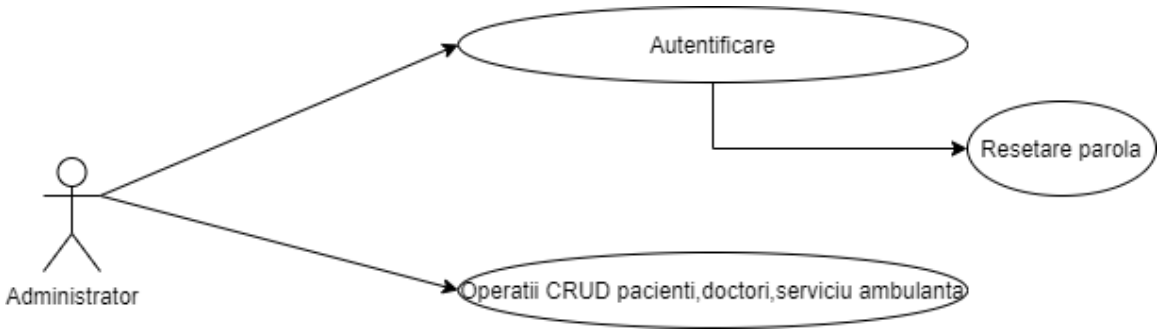


Fig 4. 1 Cazurile de utilizare ale administratorului

[7] https://firebase.google.com/docs/firestore/quotas

4.2.1.2. Serviciu de ambulanță

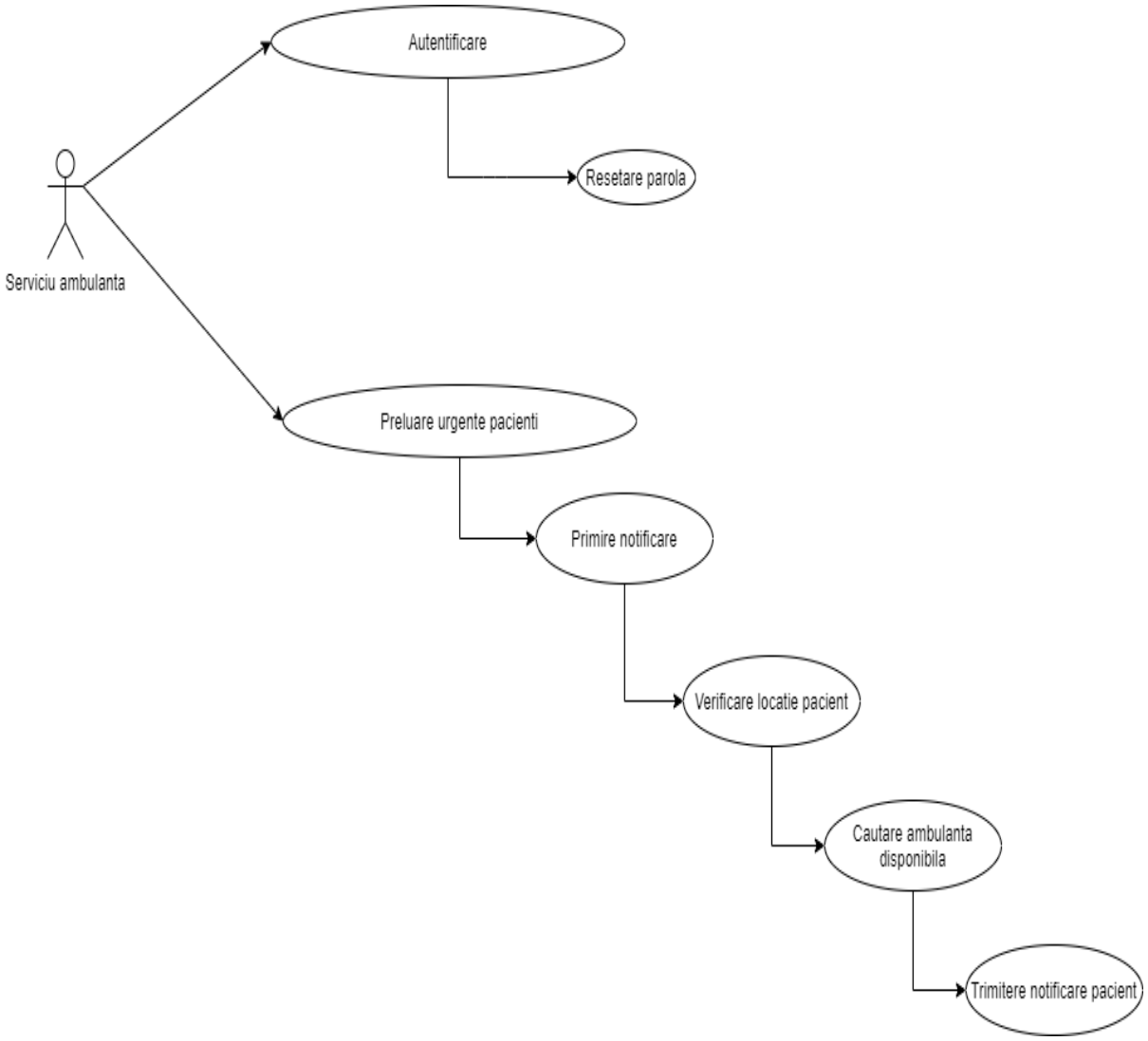


Fig 4. 2 Cazurile de utilizare ale serviciului de ambulanță

4.2.1.3. Pacient

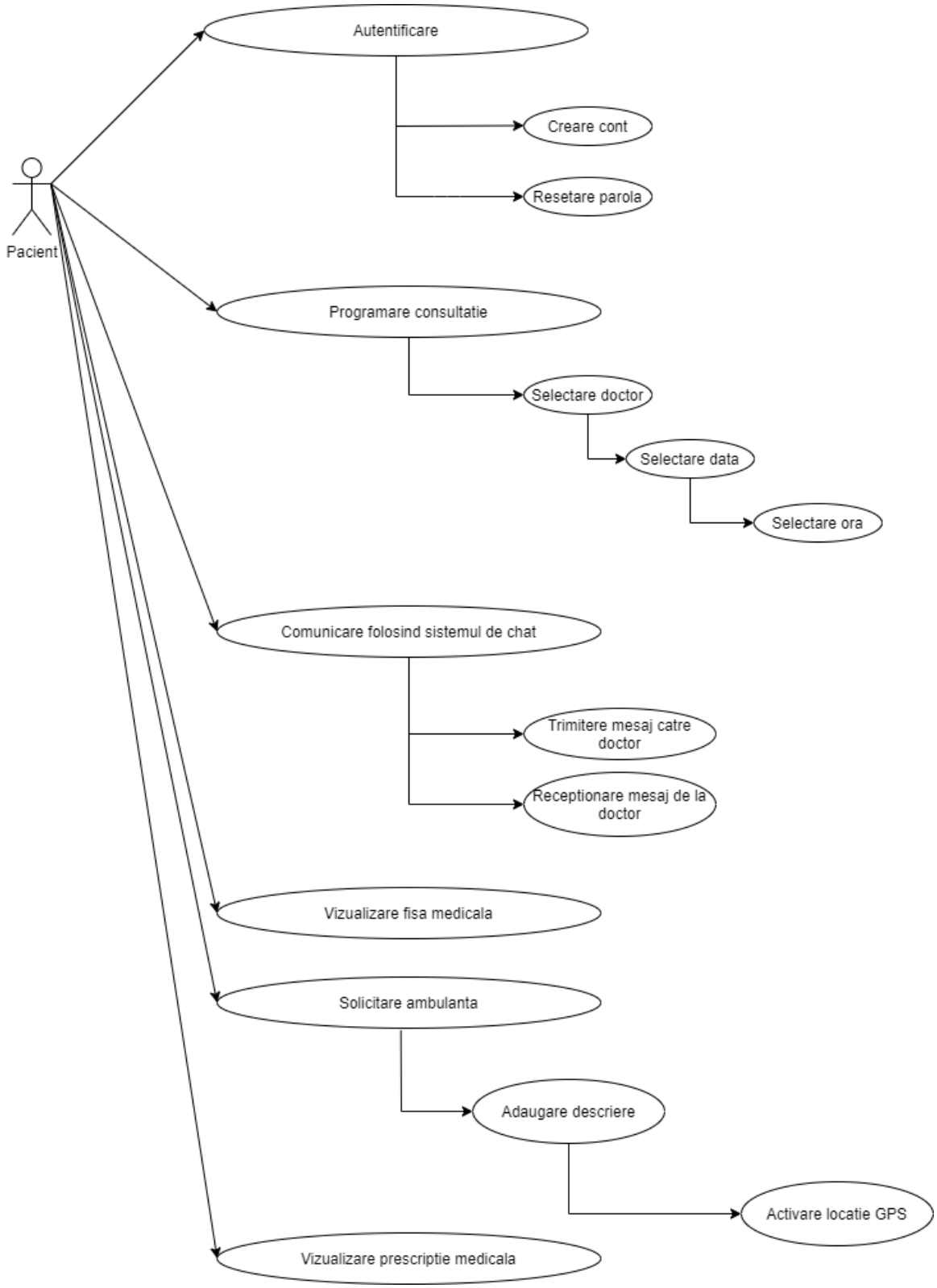


Fig 4. 3 Cazurile de utilizare ale pacientului

4.2.1.4. Doctor

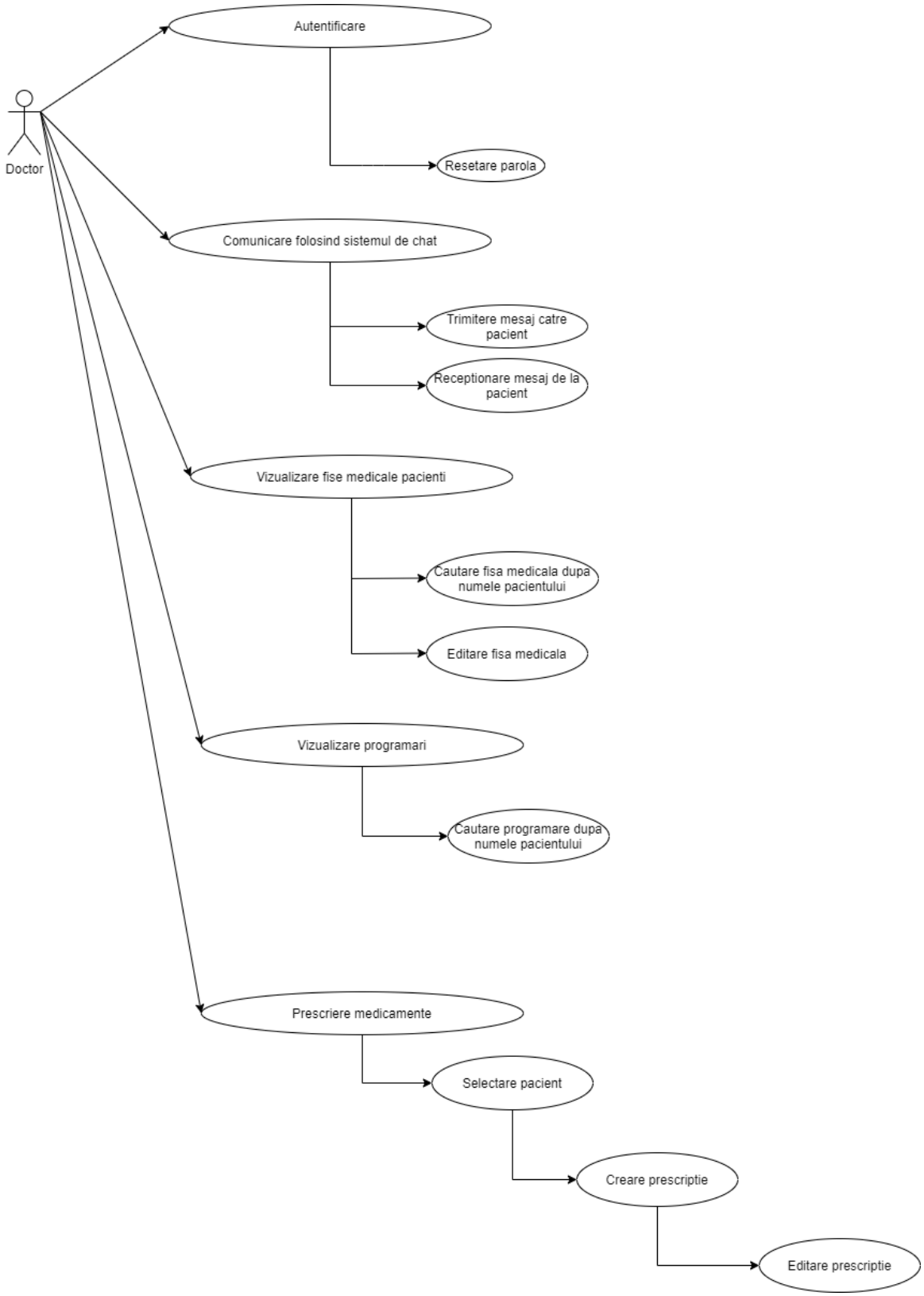


Fig 4. 4 Cazurile de utilizare ale doctorului

4.2.2. Descrierea detaliată a cazurilor de utilizare

În acest subcapitol se va realiza descrierea detaliată a cazurilor de utilizare pentru fiecare tip de actor.

4.2.2.1. Cazul de utilizare pentru autentificare

Nume: Autentificare în platformă
Actori: administrator, doctor, pacient, serviciu ambulanță
Precondiții:

1. Actorul are un cont în sistem
2. Actorul are conexiune la internet

Postcondiții:

1. Utilizatorul se autentifică cu succes
2. Utilizatorul este redirecționat către pagina proprie a aplicației

Scenariu principal:

1. Utilizatorul accesează aplicația de pe dispozitivul mobil
2. Utilizatorul introduce adresa de mail și parola
3. Utilizatorul se autentifică folosind butonul de autentificare

Scenariu alternativ:

3. Dacă adresa de mail sau parola introduse nu se află în baza de date atunci se va afișa un mesaj de eroare.

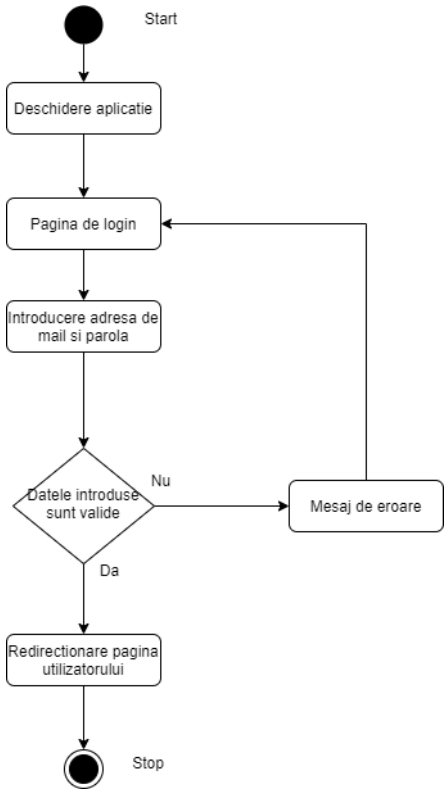


Fig 4. 5 Flowchart pentru autentificare

4.2.2.2. Cazul de utilizare pentru creare cont

Nume: Creare cont nou
Actori: pacient
Precondții:

1. Actorul nu are un cont în sistem
2. Actorul are conexiune la internet

Postcondiții:

1. Contul nou a fost creat și adăugat în sistem cu succes
2. Utilizatorul este redirecționat la pagina de autentificare a aplicației
3. Utilizatorul poate accesa aplicația folosind contul creat

Scenariu principal:

1. Utilizatorul accesează aplicația de pe dispozitivul mobil
2. Utilizatorul apasă butonul de „Creare Cont”
3. Utilizatorul introduce datele necesare nume, prenume, data nașterii, adresa, telefonul, adresa de mail și parola
4. Utilizatorul finalizează operația prin apăsarea butonului „Creare”

Scenariu alternativ:

4. Adresa de mail se află deja în baza de date a aplicației atunci se va afișa un mesaj corespunzător de genul “Adresa de mail există deja în sistem”, iar utilizatorul trebuie să folosească altă adresă de mail pentru a-și putea crea un cont și pentru a folosi aplicația.

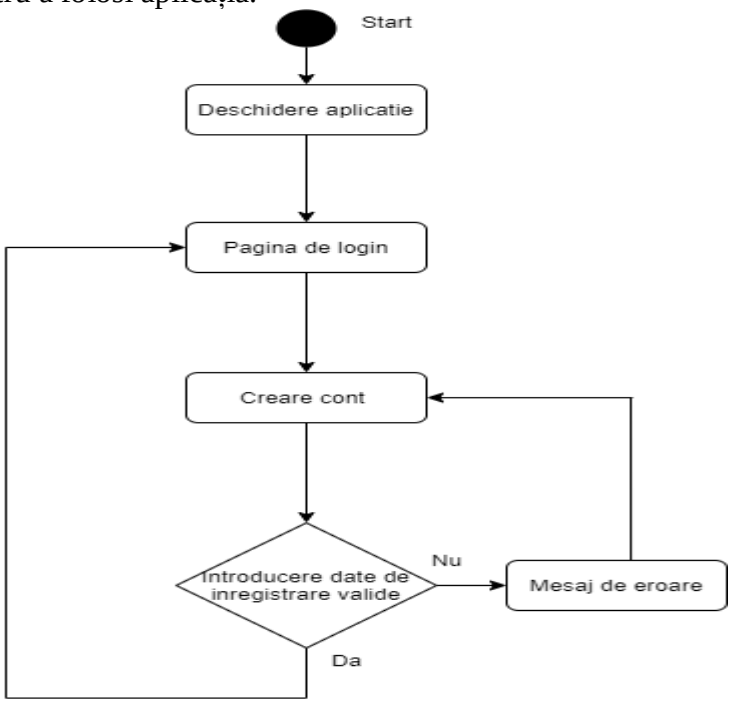


Fig 4. 6 Flowchart pentru creare cont

4.2.2.3. Cazul de utilizare pentru resetarea parolei

Nume: Resetarea parolei
Actori: administrator, pacient, doctor, serviciu de ambulanță
Precondții:

1. Actorul are un cont în sistem
2. Actorul are conexiune la internet
3. Actorul se află pe pagina de resetare a parolei care poate fi accesată de pe pagina de autentificare

Postcondiții:

1. Utilizatorul resetează cu succes parola
2. Utilizatorul se poate autentifica folosind noua parolă

Scenariu principal:

1. Utilizatorul accesează aplicația de pe dispozitivul mobil
2. Utilizatorul selectează opțiunea de Resetare parola din meniul de autentificare
3. Utilizatorul introduce adresa de mail existentă corespunzătoare contului acestuia
4. Utilizatorul primește pe adresa de mail link-ul necesar resetării parolei
5. Utilizatorul accesează acel link,iar apoi va introduce noua parolă
6. Utilizatorul se poate autentifica în aplicație folosind noua parolă

Scenariu alternativ:

3. Dacă adresa de mail introdusă nu există în baza de date se va afișa un mesaj de eroare

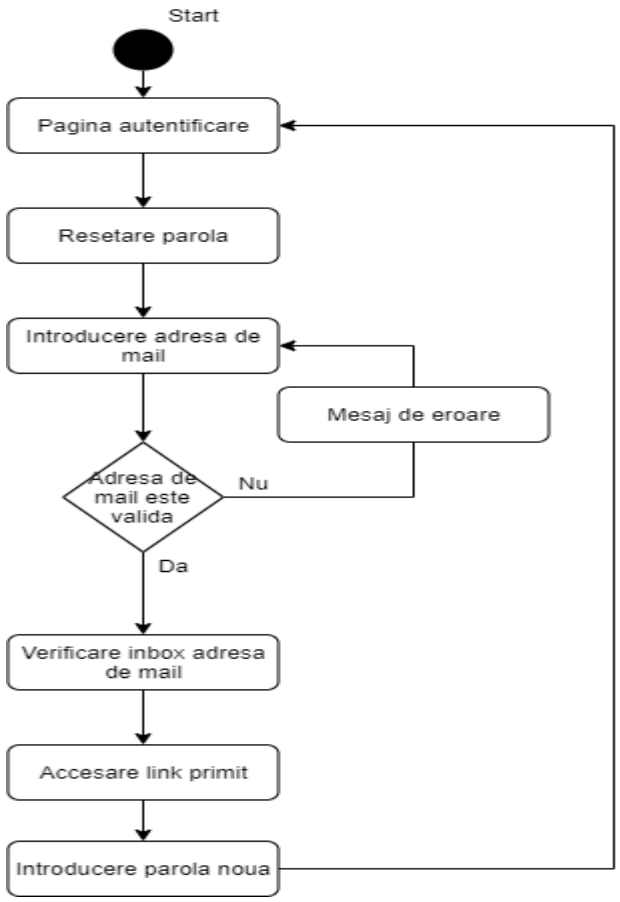


Fig 4. 7 Flowchart pentru resetarea parolei

4.2.2.4. Cazul de utilizare pentru programarea online

Nume: Programare
Actori: doctor, pacient
Precondiții:

1. Actorul are un cont în sistem
2. Actorul are conexiune la internet
3. Actorul se află pe pagina programării care poate fi accesată din meniul principal

Postdondiții:

1. Utilizatorul creează cu succes o programare
2. Nu se mai pot face programări la data și ora creată

Scenariu principal:

1. Pacient
   1.1. Pentru a crea o programare pacientul selectează doctorul la care dorește să își facă o programare
   1.2. Pacientul selectează ora si data la care dorește să își creeze programarea
   1.3. Dupa selectarea datei și orei se va afișa un mesaj de succes
   1.4. Din meniul principal se va selecta opțiunea de vizualizare programări unde se vor putea vizualiza toate programările specifice pacientului
2. Doctor
   2.1. Din meniul principal se va selecta opțiunea de vizualizare programări,iar acolo se vor putea vedea toate programările

Scenariu alternativ:

1. Pacient
   1.2. Dacă in data selectată nu sunt ore disponibile atunci trebuie aleasă altă zi
   1.3. Dacă în data selectată nu sunt programări atunci se va afișa un tabel gol pentru ziua respectivă
2. Doctor
   2.1. Dacă doctorul respectiv nu are nici o programare atunci nu se va afișa nimic in secțiunea de programări.

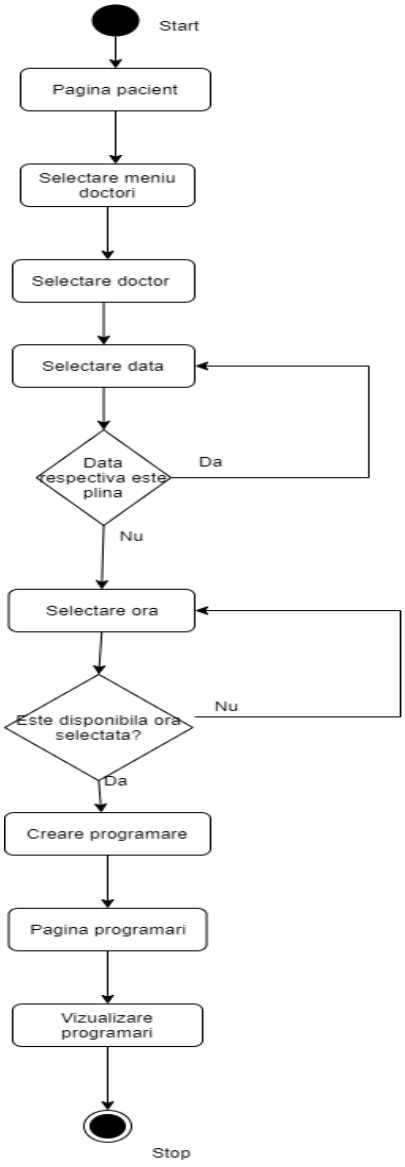


Fig 4. 8 Flowchart pentru programări

4.2.2.5. Cazul de utilizare pentru sistemul de chat

Nume: Sistem de chat

Actori: pacient, doctor

Precondiții:

1. Actorul are un cont în sistem
2. Actorul are conexiune la internet
3. Actorul se află pe pagina sistemului de chat care se poate accesa din meniul principal

Postcondiții:

1. Actorii au făcut schimb de mesaje
2. Mesajele au fost trimise cu succes

Scenariu principal:

1. Pacient

1.1. Pacientul alege doctorul cu care dorește să comunice
1.2. Pacientul scrie mesajul pe care îl trimite la doctor folosind butonul „Trimite”

2. Doctor
   2.1. Doctorul vede mesajele primite și poate răspunde la acestea trimițând alte mesaje folosind butonul „Trimite”

Scenariu alternativ:

- Dacă utilizatorul nu are o conexiune bună la internet atunci mesajele nu vor putea fi transmise sau se vor trimite foarte greu

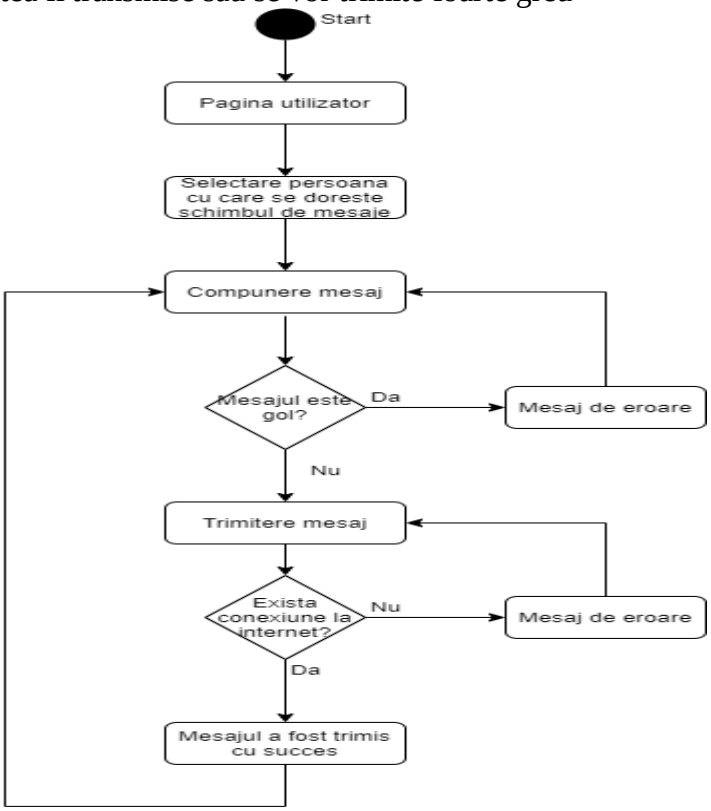


Fig 4. 9 Flowchart pentru sistemul de chat

4.2.2.6. Cazul de utilizare pentru notificări urgențe

Nume: Notificări urgențe
Actori: pacient, serviciu ambulanță
Preconditii:

1. Actorul are un cont în sistem
2. Actorul are conexiune la internet
3. Actorul se află pe pagina urgențelor care poate fi accesată din meniul principal

Postconditii:

1. Pacientul trimite o notificare către serviciul de ambulanță
2. Ambulanța care răspunde solicitării nu va fi disponibilă o perioadă de timp

Scenariu principal:

1. Pacient

1.1. Pacientul selectează butonul Urgenta medicala
1.2. Pacientul completează la descriere problema medicală pe care o are
1.3. Pacientul trimite notificarea către serviciul de ambulanță folosind butonul Trimite
1.4. Pacientul este înștiințat printr-o notificare că o ambulanță va veni la locația acestuia

2. Serviciu de ambulanță
2.1. Serviciul de ambulanță primește o notificare de la pacient. Va accesa notificarea pentru a vedea detaliile despre acesta, descrierea urgenței medicale precum și locația acestuia.
2.2. Serviciul de ambulanță care răspunde solicitării pacientului îl va înștiința printr-o notificare în care va specifica timpul necesar deplasării până la locația acestuia

Scenariu alternativ:

1. Pacient
1.2. Dacă câmpul pentru descriere nu este completat nu se va putea trimite notificarea
1.3. Dacă pacientul nu are conexiune la internet notificarea nu se va putea trimite
1.4. Dacă nici o ambulanță nu este disponibilă se va afișa un mesaj “Vă rugăm reveniți în câteva minute”
2. Serviciul de ambulanță
2.1. Dacă serviciul de ambulanță nu are conexiune la internet atunci nu va putea primi notificări de la pacienți în legătură cu urgența acestora

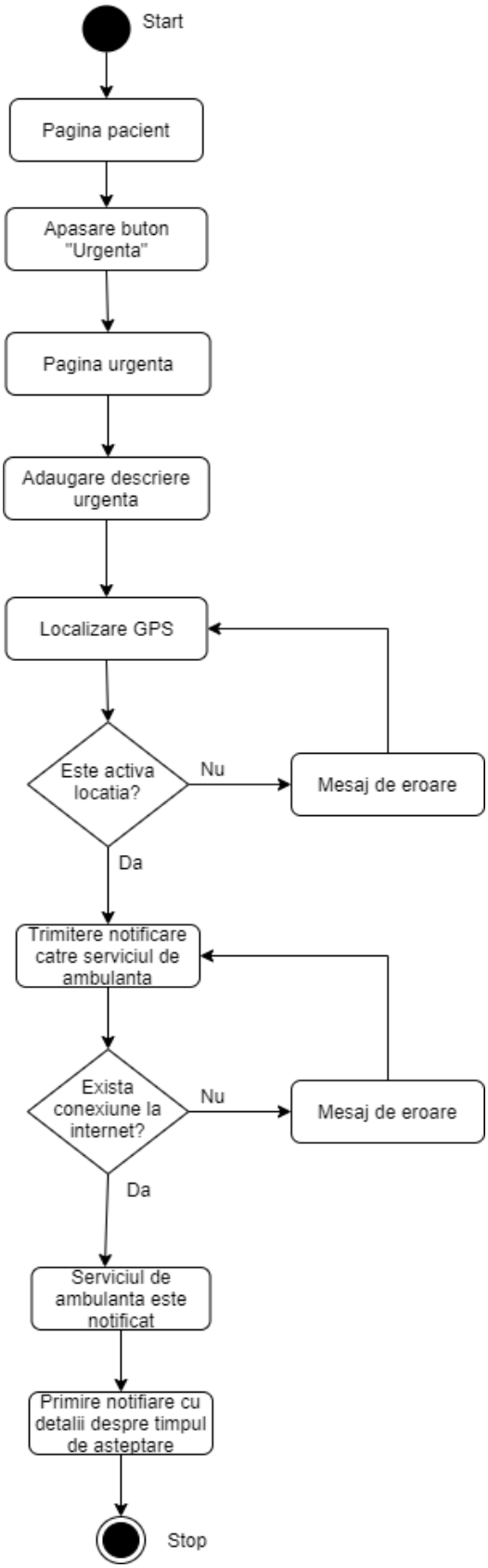


Fig 4. 10 Flowchart pentru notificări urgențe

4.2.2.7. Cazul de utilizare pentru fișa medicală

Nume: Fișă medicală
Actori: pacient, doctor
Precondiții:

1. Actorul are un cont în sistem
2. Actorul are conexiune la internet
3. Actorul se află pe pagina fișei medicale care poate fi accesată din meniul principal

Postcondiții:

1. Pacientul poate vizualiza propria fișă medicală
2. Doctorul poate crea sau edita fișa medicală a fiecărui pacient înscris în aplicație

Scenariu principal:

1. Doctor
   1.1. Doctorul selectează pacientul caruia trebuie sa îi creeze sau editeze fișa medicală
   1.2. Doctorul introduce datele despre pacient cum ar fi înălțimea,greutatea,grupa sanguină,alergii,intoleranțe,boli
   1.3. Dacă pacientul selectat are deja o fișă medicală în sistem atunci se va actualiza acea fișă
2. Pacient
   2.1. Pacientul poate vizualiza fișa medicală proprie selectând opțiunea de Fișă medicală din meniul acestuia

Scenariu alternativ:

1. Pacient
   2.1. Dacă pacientul nu are o fișă medicală în sistem atunci nu va exista informație de afișat

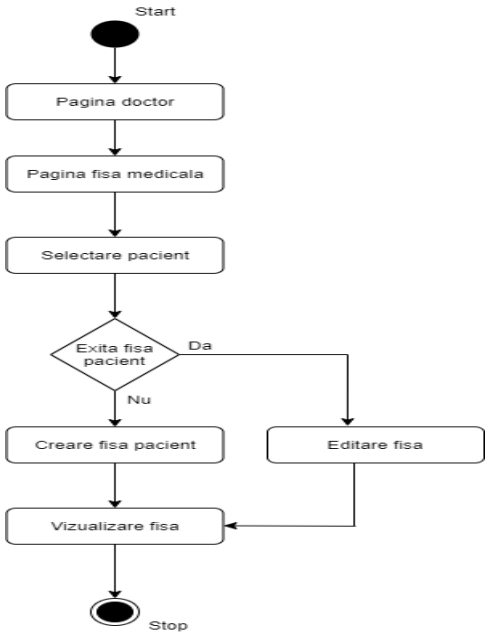


Fig 4. 11 Flowchart pentru fișa medicală

4.2.2.8. Cazul de utilizare pentru prescripție medicală

Nume: Prescripție medicală
Actori: pacient, doctor
Precondiții:

1. Actorul are un cont în sistem
2. Actorul are conexiune la internet
3. Actorul se află pe pagina prescripției medicale care poate fi accesată din meniul principal

Postcondiții:

1. Doctorul poate crea prescripții pentru pacientul înregistrați în sistem
2. Pacientul poate vizualiza toate prescripțiile create pentru acesta

Scenariu principal:

1. Doctor
   1.1. Doctorul selectează pacientul căruia trebuie să îi creeze o prescripție medicală
   1.2. Doctorul introduce datele despre prescripție cum ar fi numele medicamentului, efecte secundare, administrarea acestuia și prețul
   1.3. Doctorul adaugă cu succes o prescripție medicală pentru pacientul respectiv
2. Pacient
   2.1. Pacientul poate vizualiza toate prescripțiile medicale care îi sunt destinate

Scenariu alternativ:

1. Doctor
   1.2. Dacă doctorul nu completează toate datele atunci nu se va putea crea o prescripție pentru pacient
2. Pacient
   2.1. Dacă pacientul nu are nici o prescripție medicală atașată acestuia atunci nu va vizualiza nimic în fereastra de prescripție

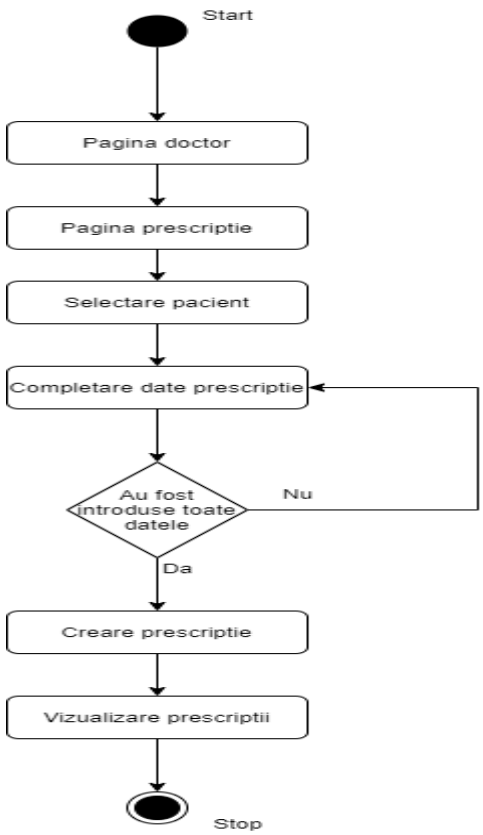


Fig 4. 12 Flowchart pentru prescripție medicală

4.3. Tehnologii utilizate

4.3.1. Firebase

Firebase este o platformă de cloud oferită de Google care oferă o varietate de tool-uri și servicii pentru crearea aplicațiilor web și mobile. Această platformă este de tipul BaaS(Backend-as-a-Service). Un avantaj al utilizării Firebase este că aceasta suportă mai mulți utilizatori înregistrați în acelasi timp, iar serviciile sunt disponibile non-stop. Această platformă dispune de mai multe componente în funcție de nevoile dezvoltatorului. Vor fi detaliate doar acele componente care au fost utilizate în dezvoltarea acestei aplicații:

4.3.1.1. Firebase Authentication

Pentru partea de autentificare și înregistrare s-a folosit componenta Firebase Authentication[8]. Această componentă este specializată în stocarea datelor de autentificare a utilizatorilor. Aceste date vor putea fi folosite de utilizatori atunci când doresc să se autentifice în aplicație. În cazul în care utilizatorul nu își mai amintește parola, sistemul poate trimite către utilizator un email pentru resetarea parolei. Datele stocate în Firebase Authentication sunt securizate folosind un Token de autentificare.

[8] https://firebase.google.com/docs/auth

Firebase Authentication dispune de mai multe moduri de autentificare, printre acestea se numară și autentificare folosind contul unor aplicații cunoscute cum ar fi Facebook, Twitter sau Google. De asemenea este permisă autentificarea folosind numărul propriu de telefon. Mai mult Firebase Authentication mai dispune și de o librărie pentru interfața grafică și anume FirebaseUI Auth, care conține mai multe pagini în care este integrată componenta de autentificare.

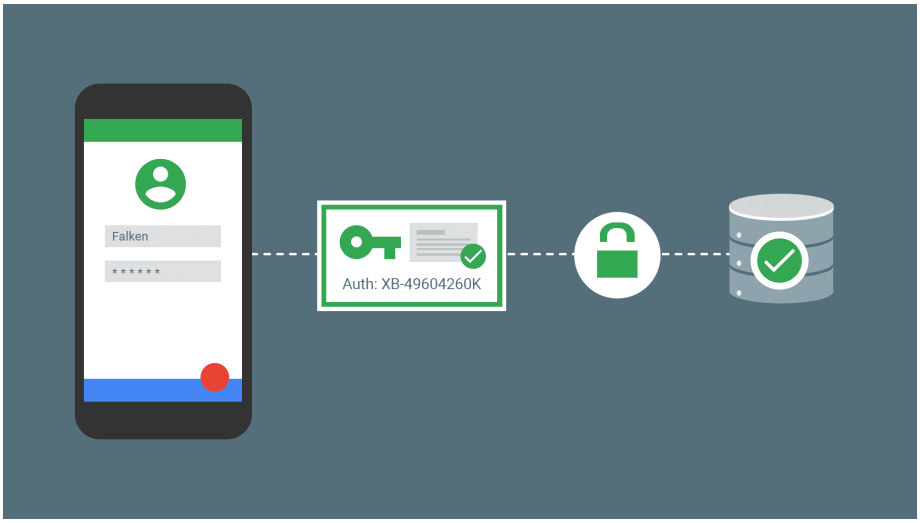


Fig 4. 13 Firebase Authentication

4.3.1.2. Firestore Database

Pentru a stoca datele din aplicație am folosit Firestore Database[9]. Aceasta este o bază de date non-relațională, care permite accesul și modificarea datelor de către utilizatori într-un mod concurent. Datele sunt stocate în documente de tip JSON. Acestea sunt ușor de accesat și de modificat. Aceste documente pot fi la rândul lor grupate într-o colecție pentru o mai bună structurare a datelor. Astfel pot exista mai multe colecții care conțin mai multe documente.

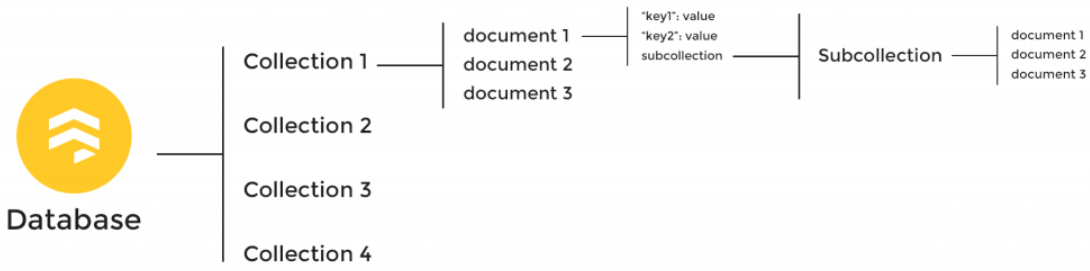


Fig 4. 14 Structura de date Firestore Database[10]

[9] https://firebase.google.com/docs/firestore
[10] https://waelyasmina.com/firebase-cloud-firestore-tutorial-web/

4.3.1.3. Cloud Messaging

Firebase Cloud Messaging[11] este o soluție de mesagerie care permite trimiterea de mesaje sau notificări în mod fiabil, fără costuri. Folosind această funcționalitate a Firebase-ului se poate notifica o aplicație client în legătură cu diferite cereri cum ar fi mesaje de conținut pentru reținerea utilizatorilor. În cazul mesageriei un mesaj poate transfera o încărcătură utilă de până la 4kB către o aplicație client. Mai mult, mesajele către clienți pot fi trimise în trei moduri: către un anumit dispozitiv, către un grup de dispozitive sau către toate dispozitivele.

4.3.2. Android

Android este un sistem de operare destinat dispozitivelor mobile cu touchscreen cum ar fi smarthphone-urile și tabletele. A fost dezvoltat de cei de la Open Handset Alliance și sponsorizat de cei de la Google. Este un software gratuit care se bazează pe limbajul de programare Java.

Este cel mai bine vândut sistem de operare pentru dispozitive mobile din 2011 și 2013 pentru tablete. Din luna mai a acestui an există peste trei miliarde de utilizatori activi în fiecare lună, cea mai mare bază instalată a oricărui sistem de operare, reprezentând 71% din numărul tuturor smartphone-urilor existente[12].

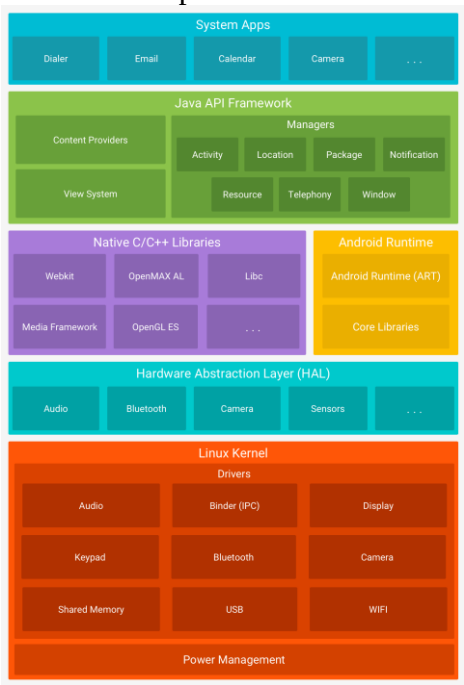


Fig 4. 15 Arhitectura Android[13]

[11] https://firebase.google.com/docs/cloud-messaging/

[12] https://www.businessofapps.com/data/android-statistics/

[13] https://developer.android.com/guide/platform

4.3.3. Google Maps

Pentru ca pacientul să poată furniza locația exactă a acestuia s-a folosit API-ul Google Maps oferit de cei de la Google. Acest API se folosește pentru a prelua locația curentă a pacientului ca să poată fi trimisă mai departe către serviciul de ambulanță astfel încat acesta să poată ajunge într-un timp cât mai scurt la pacient.

Capitolul 5. Proiectare de Detaliu si Implementare

În acest capitol se vor prezenta tipul de arhitectură folosit pentru sistem, diagramele de pachete, de clase, diagrama bazei de date, precum și implementarea detaliată a funcționalităților care va fi împărțită pe module.

5.1. Arhitectura sistemului

Fig 5. 1 Arhitectura sistemului

Arhitectura sistemului este una de tip client-server. În acest tip de arhitectură avem ca și clienți utilizatorii aplicației care mai pot fi numiți și consumatori, iar serverul poate fi denumit și producător. Aceste două componente comunică între ele. Serverul este format din baza de date, care gestionează resursele și serviciile aplicației care sunt solicitate de către client. Clientul este format din interfața pe care o folosesc utilizatorii pentru a comunica cu serverul. Această arhitectura se bazează pe un mecanism de cerere si răspuns. Utilizatorul trimite o cerere către server, iar acesta va furniza prin intermediul unui răspuns datele solicitate de către client. Pentru comunicarea dintre cele două resurse este necesară o conexiune la internet.

Componentele sistemului:

- **Client:** sunt dispozitivele mobile care sunt folosite de către utilizatori și anume administrator, pacienți, doctori și serviciul de ambulanță pentru a trimite solicitări către server
- **Server:** este baza de date, în acest caz Firebase, care este un serviciu de cloud oferit de cei de la Google și Google Maps. Aceste servere primesc cereri de la utilizatorii dispozitivelor mobile și furnizează răspunsuri pentru aceste cereri.

5.2. Modelul bazei de date

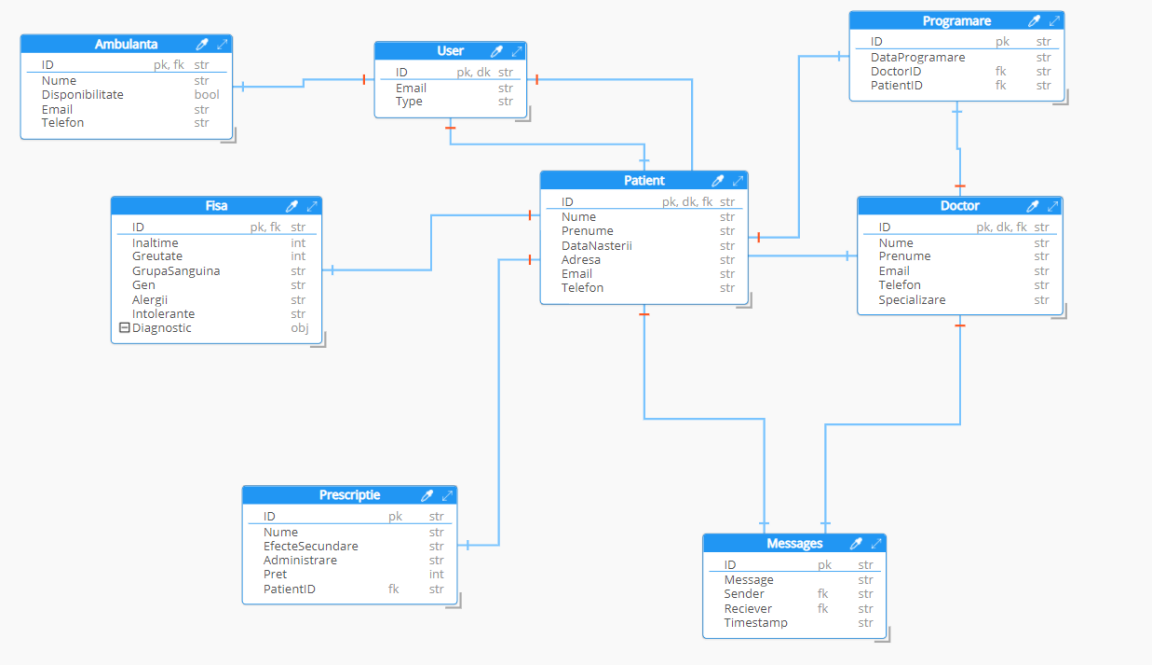


Fig 5. 2 Diagrama bazei de date

În figura de mai sus este prezentată schema bazei de date care a fost realizată folosind Hackolade[14]. Baza de date utilizată pentru această aplicație este Firebase Cloud Firestore.

Cloud Firestore este o bază de date non-relațională, oferita de Google care oferă suport pentru aplicații Web, Android sau iOS prin intermeniul unor SDK native. Conexiunea dintre baza de date și IDE-ul de lucru Android Studio a fost realizată folosind asistența Firebase care este încorporată în IDE.

Mai jos vor fi prezentate fiecare colecție si ce semnifică câmpurile documentelor. Fiecare document din colecții este caracterizat de către un ID propriu și unic care este generat în momentul creării documentului respectiv.

- Pacient: conține numele, prenumele, adresa, dataNasterii, adresa de mail, telefon, toate sunt de tipul String
- Doctor: conține numele, prenumele, telefon, adresa de mail și specializare, toate de tipul String
- Serviciu de ambulanță: conține nume, adresa de mail și telefon, de tip String și mai conține și disponibilitate care este boolean
- User: conține adresa de mail, parola și tipul utilizatorului, toate de tip String
- Programare: conține ID-urile doctorului și al pacientului, de tip String precum și data creării programării, de tip timestamp
- Mesaje: conține ID-urile celui care trimite mesajul și celui care recepționează mesajul precum și conținutul mesajului, de tip String și mai conține momentul în care a fost creat mesajul, de tipul timestamp

[14] https://hackolade.com/

- Fișă:conține înălțimea și greutatea de tip number, ID-ul pacientului, gen, grupa sanguină, alergii, intoleranțe de tip String și mai conține diagnostic care este un Map de String și String, reprezentând diagnosticul și data la care a fost stabilit acesta
- Prescripție: conține ID-ul pacientului, numele medicamentului, efectele secundare ale acestuia, administrarea, toate de tip String și prețul care este de tip int

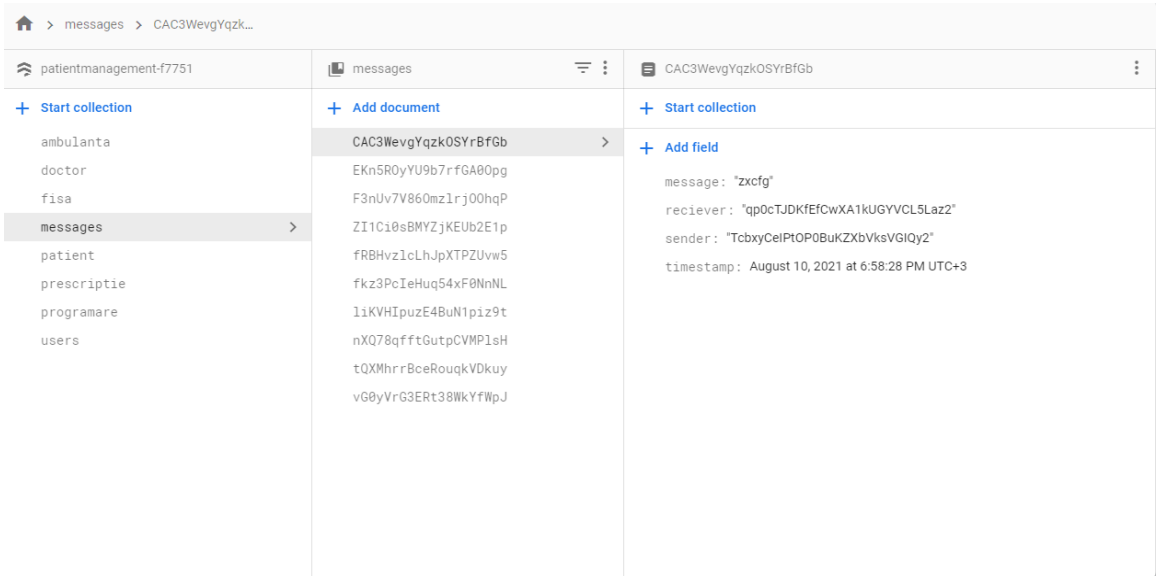


Fig 5. 3 Baza de date Firestore

5.3. Diagrama de pachete

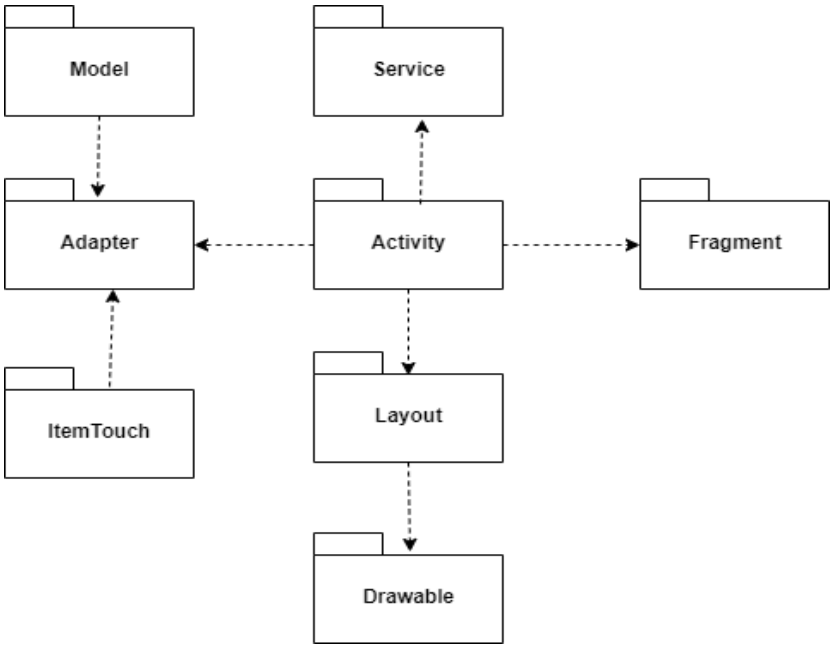


Fig 5. 4 Diagrama de pachete

În figura 5.4 este prezentată diagrama de pachete a aplicației. Pachetele sunt denumite în acest mod pentru a delimita funționalitățile.

- Layout-definesc structura interfeței aplicației
- Drawable-conține imaginile care se regăsesc în aplicație
- Models-conține clasele care definesc obiecte
- Adapters-se ocupă cu gestionarea accesului la datele care se află în componentele de view și de afișarea lor
- ItemTouch-sunt folosite pentru a adăuga funcționalitatea de swipe, drag & drop unui RecycleView
- Fragment-reprezintă o porțiune reutilizabilă din interfața aplicației. Este definit de o activitate, dar are layout propriu și se ocupă singur de evenimente.
- Activities-se ocupă cu crearea de pagini cu care interacționează utilizatorul. În timp ce layout-ul definește partea de interfață a unei pagini, activities definește modul de funcționare a paginii

5.4. Diagrama de clase

În aplicațiile Android fiecare fereastră este reprezentată de către o activitate. În anumite situații pentru o eficiență mai bună se pot utiliza fragmente pentru actualizarea unor date de pe o porțiune din activitate. Pentru rularea oricărei aplicații Android este nevoie de cel puțin o activitate.

MainActivity este activitatea principală din aplcație, astfel că aceasta gestionează în mod direct sau indirect celelalte activități din cadrul sistemului. Aceasta conține fereastra de autentificare de unde fiecare utilizator va fi redirecționat către pagina lui în funcție de tipul acestuia.

Clasele din pachetul *Model* sunct clasele responsabile de crearea obiectelor. Aceste clase sunt *PatientModel, DoctorModel, ServiciuAmbulantaModel, PrescriptieModel, FisaMedicalaModel, ProgramareModel, MesajModel, PageViewModel.*

Clasele din pachetul *Adapter* sunt clasele responsabile pentru legătura dintre **RecycleView** și obiectele din baza de date. Cu ajutorul unui RecycleView se realizează popularea cu date de la server a interfeței grafice. Clasele din acest pachet sunt *PatientAdapter, DoctorAdapter, ServiciuAmbulantaAdapter, PatientAdapterMesaj, DoctorAdapterMesaj, MessageAdapter, SectionPageAdapter.*

Clasele din pachetul *Fragment* sunt clasele responsabile pentru managementul celorlalți utilizatori. Aceste fragmente fac parte din clasa de activități *Admin*, fiecare fragment din această activitate este destinată pentru tipurile de utilizatori și anume pacient, doctor și serviciu de ambulanță. Aceste clase sunt *PatientFragment, DoctorFragment, ServiciuAmbulantaFragment, PlaceholderFragment.*

Clasele din pachetul *ItemTouch* sunt clasele responsabile pentru adăugarea opțiunii de swipe left și right, pentru obiectele afișate de un RecycleView. Pentru ștergerea sau editarea datelor celorlalți utilizatori din RecycleView-ul acestora se va alege una din aceste operații folosind swipe left pentru editare și swipe right pentru ștergere. Aceste clase sunt *TouchHelper, TouchHelperDoctor, TouchHelperServiciuAmbulanta.*

Clasele din pachetul *Activity* sunt clasele responsabile de interacțiunea utilizatorului cu aplicația. Pentru partea de interfață acestea sunt în strânsă legătură cu fișierele de **Layout**, pentru afișarea datelor, dar și pentru prelucrearea acestora. Aceste clase care implementează activități sunt *AddDoctor, AddPatient, AddServiciuAmbulanta, Admin, DoctorInterface, PatientInterface, ServiciuAmbulantaInterface, ListaPacienti, ListaPacientiFisa, CalendarProgramare, FisaMedicala, ListaDoctori, ListaDoctoriMesaj, ListaDoctoriProgramare, Mesaj, UrgentaMedicala, Register, ResePassword, MainActivity.*

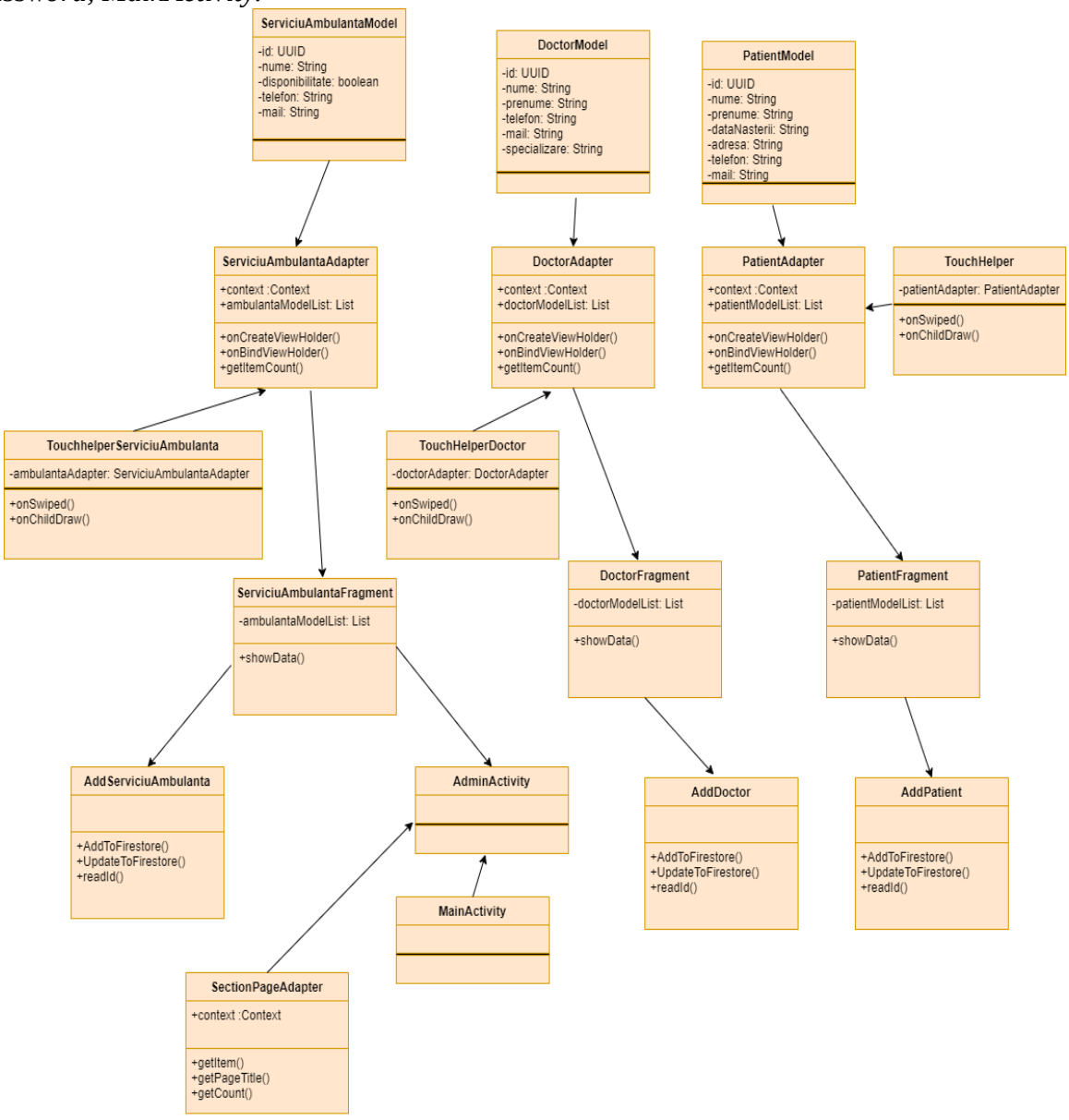


Fig 5. 5 Diagrama de clase pentru modulul administratorului

După auntetificare care este realizată pe pagina principală și anume MainActivity, administratorul este redirecționat pe pagina Admin, care reprezintă interfața administratorului. Aceasta este compusă din trei fragmente, fiecare din acestea destinat pentru un tip de utilizator și anume pacient, doctor și serviciu de ambulanță. Secționarea

paginii în cele trei fragmente este realizată cu ajutorul clasei SectionPageAdapter. Fiecare din aceste fragmente conține un RecycleView cu listele utilizatorilor. Afișarea datelor în RecycleView este realizată cu ajutorul claselor Adapter în care se specifică ce date ale claselor Model se vor afișa. De asemenea mai sunt clasele TouchHelper care implementează funcția de swipe asupra elementelor din listă. Pentru adăugarea de utilizatori noi din fragmentele respective se va deschide o fereastră nouă, pentru adăugarea unui utilizator. Acolo vor trebui completate câmpurile specificate, iar apoi utilizatorul creat va fi adăugat în sistem, fiind vizibil în lista fragmentului respectiv.

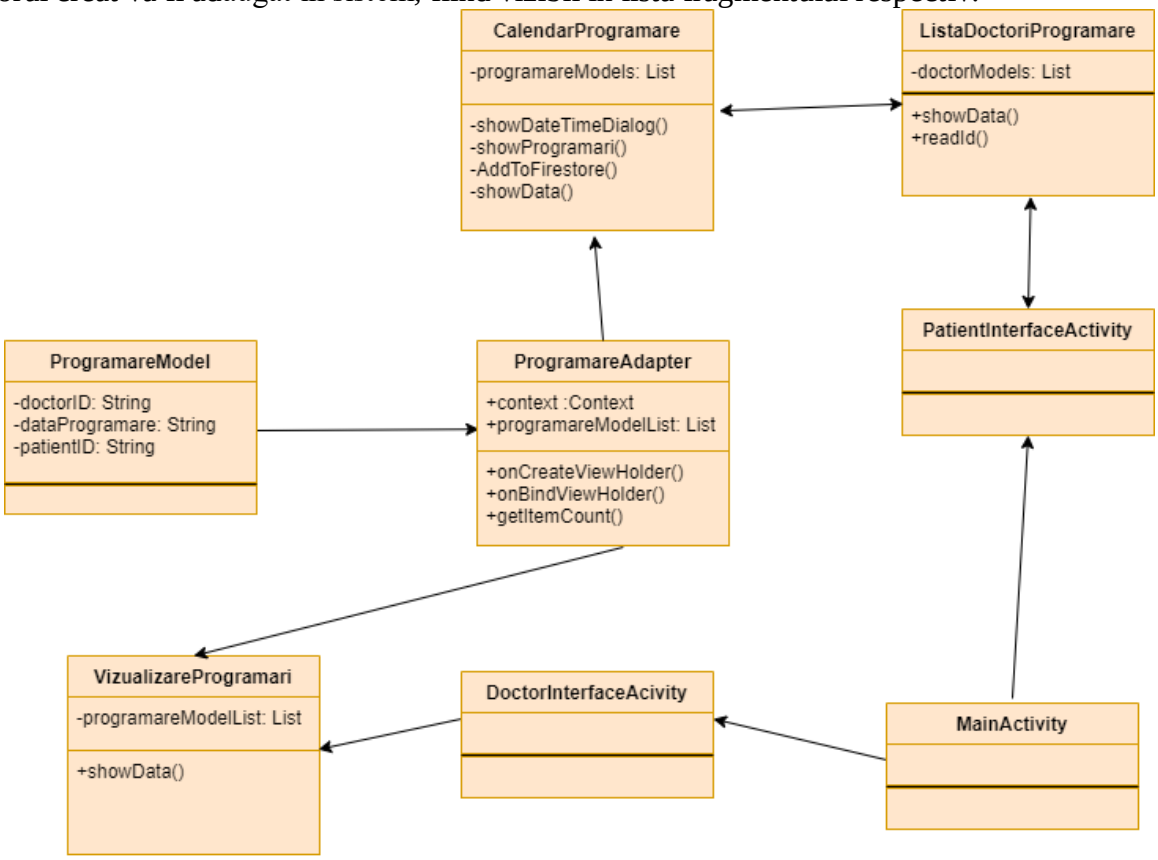


Fig 5. 6 Diagrama de clase pentru modulul de programări

Pentru a crea o programare din activitatea PatientInterfaceActivity se va apăsa butonul de Programare online și se va deschide o fereastră nouă ListaDoctoriProgramare de unde pacientul selectează doctorul la care dorește să își creeze o programare. Apoi se va deschide pagina de programare care este implementată de clasa CalendarProgramare. În această clasă pacientul poate crea o programare și poate vizualiza programările deja existente pentru doctorul selectat. Pentru afișarea programărilor este folosit un adapter implementat în clasa ProgramareAdapter.

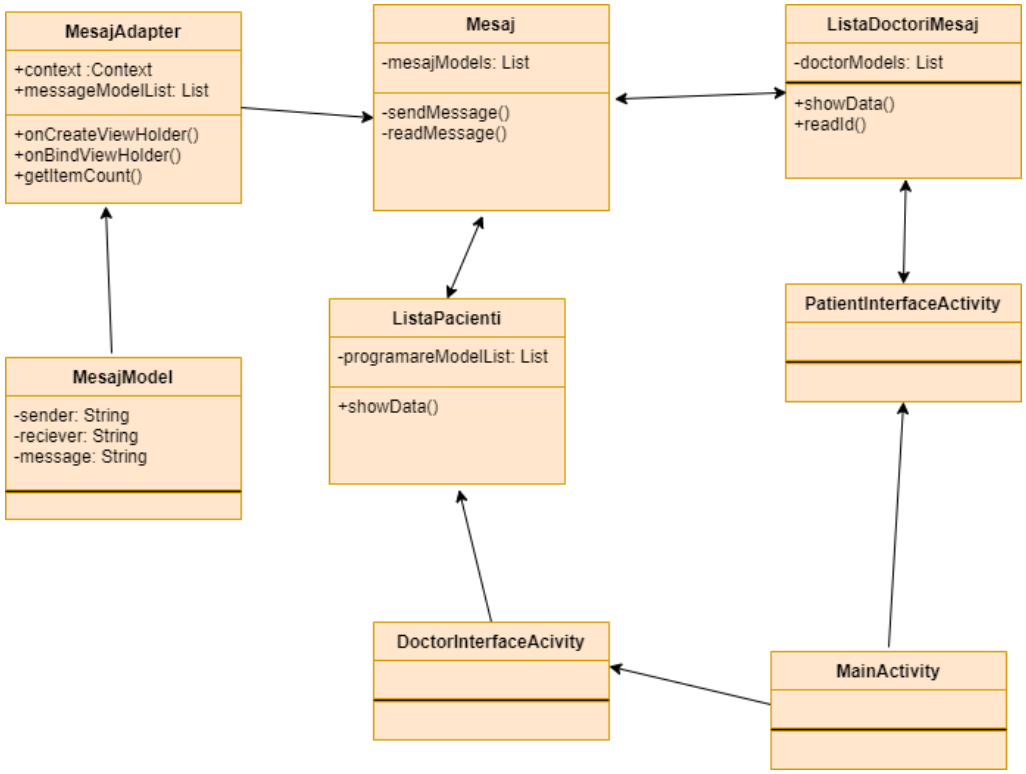


Fig 5. 7 Diagrama de clase pentru modulul de chat

Pentru citirea și trimiterea mesajelor se folosește clasa Mesaj care trimite mesaje de la utilizatorul auntentificat către utilizatorul selectat, doctor sau pacient, și citește mesajele pe care utilizatorul înregistrat le are cu ceilalți utilizatori. Mesajele sunt afișate folosind adaptorul MesajAdapter. În clasa MesajAdapter se verifică dacă utilizatorul este cel care a trimis un mesaj sau a primit, astfel dacă este emițător atunci mesajele acestuia vor fi afișate în partea dreaptă a ferestrei, iar dacă este receptor mesajele vor fi afișate în partea stângă a ferestrei.

În funcție de tipul de utilizator care se autentifică folosind MainActivity va fi redirecționat către DoctorInterfaceActivity dacă este utilizator de tip doctor sau către PatientInterfaceActivity dacă este utilizator de tip pacient.

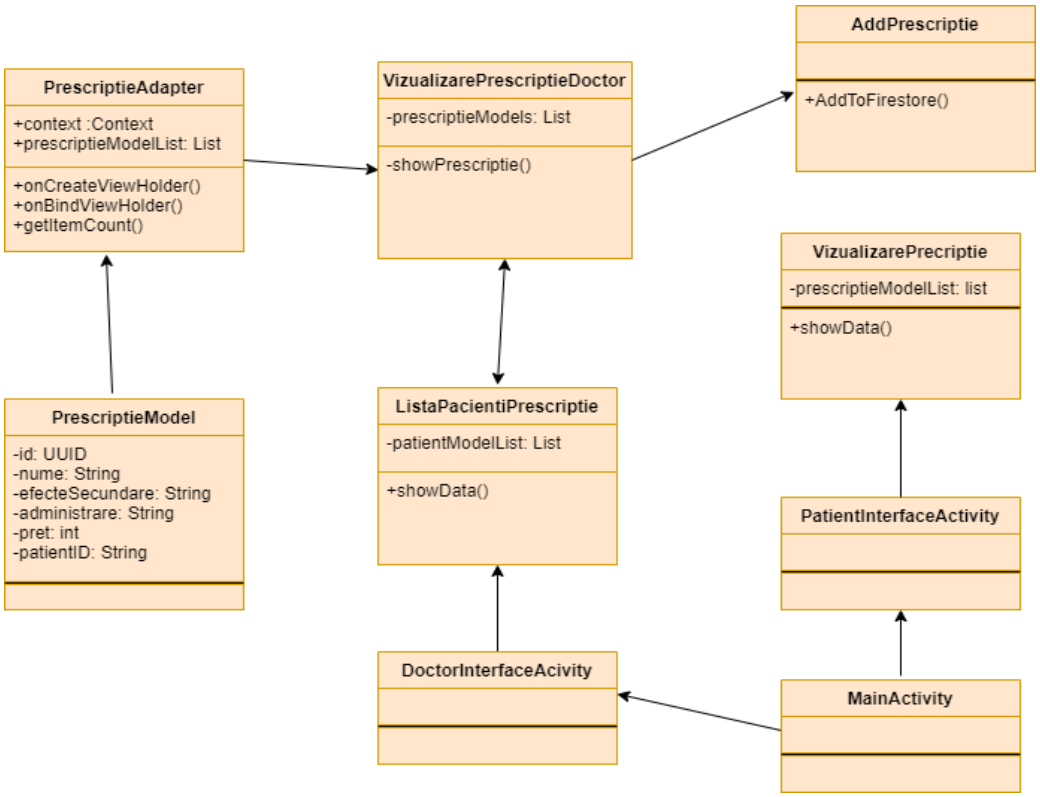


Fig 5. 8 Diagrama de clase pentru modulul prescripției

Pentru prescripțiile medicale avem clasa VizualizarePrescriptieDoctor acolo unde doctorul poate vedea prescripțiile unui pacient selectat în clasa ListaPacientiPrescriptie, iar dacă dorește să adauge o nouă prescripție se va folosi clasa AddPrescriptie. Prescripțiile sunt afișate folosind adapterul PrescriptieAdapter.

La fel ca și la programări utilizatorul va fi redirecționat către pagina de doctor sau pacient în funcție de tipul acestuia.

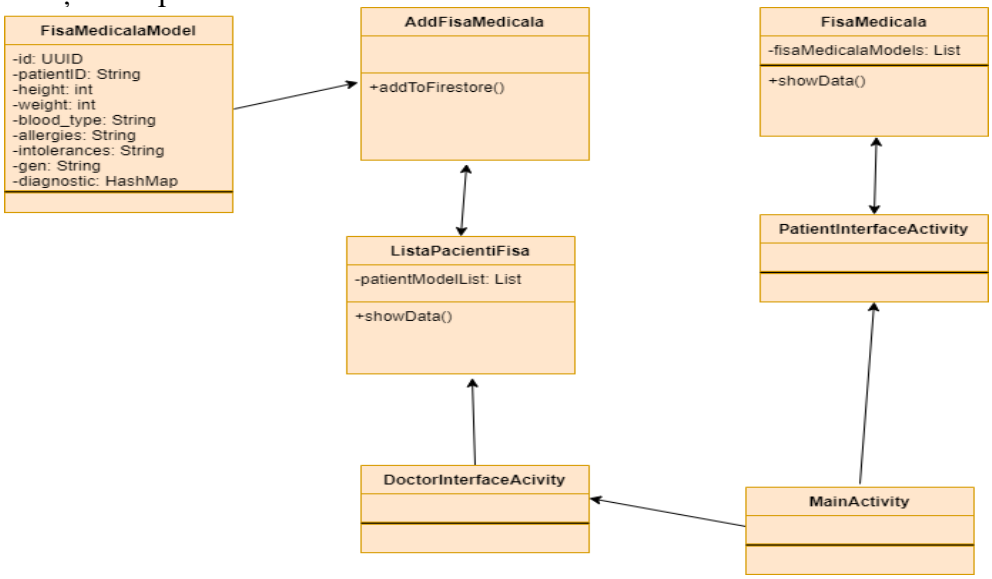


Fig 5. 9 Diagrama de clase pentru modulul fișei medicale

Pentru fișa medicală doctorul are posibilitatea de a crea una nouă pentru pacient dacă acesta nu are deja una în sistem sau o poate edita pe cea existentă. Fișa fiind un obiect unic pentru fiecare pacient nu vom mai avea un adapter care să realizeze legătura dintre activitate și model. Pacientul dupa autentificare are posibilitatea de a-și vedea fișa proprie din meniul principal din clasa PatientInterfaceActivity.

5.5. Descrierea modulelor aplicației

5.5.1. Autentificare

Pentru autentificarea în aplicație este necesară introducerea adresei de mail și parola aferente contului utilizatorului. După apăsarea butonului de „Autentificare” se va verifica în baza de date că adresa de mail și parola există. Dacă una din acestea nu este corectă se va afișa un mesaj în partea de jos a ecranului în care se va specifica că adresa de mail sau parola nu sunt corecte. Dacă acestea sunt corecte atunci utilizatorului îi va fi asignat un token cu care va putea avea autorizație pentru efectuarea anumitor operații cum ar fi adăugare, citire. Token-ul este valabil până când utilizatorul se va deconecta de pe cont.

Token-ul este reprezentat ca un ID unic care este generat aleator. În momentul în care un utilizator se autentifică va primi un alt token, indiferent de tipul acestuia. Asignarea token-ului este asigurată de către Firebase Authentication pentru orice cont existent în baza de date.

```
fAuth.signInWithEmailAndPassword(mail,parola).addOnCompleteListener(new OnCompleteListener<AuthResult>() {
    @Override
    public void onComplete(@NonNull Task<AuthResult> task) {
        if(task.isSuccessful()){
            FirebaseFirestore ref=FirebaseFirestore.getInstance();
            CollectionReference userRef=ref.collection( collectionPath: "users");
            String uid=FirebaseAuth.getInstance().getCurrentUser().getUid();
            userRef.document(uid).get().addOnCompleteListener(new OnCompleteListener<DocumentSnapshot>() {
                @Override
                public void onComplete(@NonNull Task<DocumentSnapshot> task) {
                    if(task.isSuccessful())
                    {
                        DocumentSnapshot document=task.getResult();
                        if(document.exists())
                        {
                            String type=document.getString( field: "type");
                            if(type.equals("doctor"))
                            {
                                Toast.makeText( context: MainActivity.this, text: "Logged in Succesfully", Toast.LENGTH_SHORT).show();
                                startActivity(new Intent(getApplicationContext(), DoctorInterface.class));
                            }
                            else if(type.equals("patient"))
                            {
                                Toast.makeText( context: MainActivity.this, text: "Logged in Succesfully", Toast.LENGTH_SHORT).show();
                                startActivity(new Intent(getApplicationContext(), PatientInterface.class));
                            }
                            else if(type.equals("admin"))
                            {
                                Toast.makeText( context: MainActivity.this, text: "Logged in Succesfully", Toast.LENGTH_SHORT).show();
                                startActivity(new Intent(getApplicationContext(), Admin.class));
                            }
```

Fig 5. 10 Autentificare

5.5.2. Creare cont

Crearea unui cont nou este o funcționalitate destinată exclusiv pacienților, pentru doctori și serviciul de ambulanță se va ocupa administratorul.

Pacientul care dorește să folosească aplicația, dar nu dispune de un cont are posibilitatea de a-și crea unul selectând butonul de Creare cont din meniul de autentificare. Va fi redirecționat pe pagina de creare cont unde va trebui să completeze datele necesare creării unui cont, acestea fiind nume, prenume, data nașterii, adresa, telefon, adresa de mail și parola. Dacă unul din aceste câmpuri nu este completat se va afișa un mesaj de avertizare în dreptul câmpului necompletat. Dacă adresa de mail există deja în baza de date a sitemului atunci nu se va putea crea cont nou, utilizatorul trebuie să introducă o nouă adresă de mail. Dacă toate datele introduse sunt validate, atunci se creează un cont nou de pacient. Se va adăuga în baza de date un cont atât la pacient, cât și la user. Se adaugă și la user deoarece acolo se adaugă adresa de mail și tipul de utilizator pentru toți utilizatorii existenți în aplicație. Se folosește această colecție deoarece este mai simplu de căutat un utilizator pentru a putea vedea tipul acestuia și pentru a putea fi redirecționat pe pagina acestuia.

5.5.3. Resetare parolă

În cazul în care unul dintre utilizatori nu își mai amintește parola, acesta are posibilitatea de a o reseta. Utilizatorul are nevoie doar de adresa de mail. De pe pagina de autentificare va selecta opțiunea de resetare parolă și va fi redirecționat pe pagina destinată resetării parolei. Acolo va trebui să introducă adresa de mail specifică contului acestuia. Dacă nu introduce o adresă de mail sau adresa de mail introdusă există în baza de date se va afișa un mesaj de atenționare în partea de jos a ecranului. Dacă adresa de mail introdusă este validată atunci pe acea adresă se va trimite un mail cu un link pe care utilizatorul va trebui să îl acceseze. Va fi redirecționat pe o pagină nouă unde îi va apărea un câmp unde trebuie să introducă noua parolă. După ce introduce noua parolă va primi un mesaj de suces în care se va specifica că se poate autentifica în aplicație cu parola nouă. Funcția de trimitere mail pentru resetarea parolei este o funcție existentă din pachetul Firebase, mai specific FirebaseAuthentication.

```
private void resetPassword(){
    String email=emailEditText.getText().toString().trim();
    if(email.isEmpty())
    {
        emailEditText.setError("Introduceti adresa de mail");
        emailEditText.requestFocus();
        return;
    }
    if(!Patterns.EMAIL_ADDRESS.matcher(email).matches())
    {

        emailEditText.setError("Introduceti o adresa de mail valida");
        emailEditText.requestFocus();
        return;
    }


    fAuth.sendPasswordResetEmail(email).addOnCompleteListener(new OnCompleteListener<Void>() {
        @Override
        public void onComplete(@NonNull Task<Void> task) {
            if(task.isSuccessful())
            {
                Toast.makeText( context: ResetPassword.this, text: "Verificati mailul pentru resetarea parolei",Toast.LENGTH_LONG).show();
                //startActivity(new Intent(getApplicationContext(), MainActivity.class));
            }
            else
            {
                Toast.makeText( context: ResetPassword.this, text: "A aparut o eroare.Incercati din nou",Toast.LENGTH_LONG).show();
            }
        }
    });
```

Fig 5. 11 Resetare parolă

5.5.4. Management utilizatori

Administratorul este cel care se ocupă de managementul celorlalți utilizatori. Există un singur administrator în cadrul aplicației.

Acesta poate adăuga, șterge sau edita datele celorlalți utilizatori. În cazul doctorilor și al serviciului de ambulanță el se ocupă de crearea conturilor, iar în cazul pacienților se poate ocupa în cazul în care pacienții care doresc să folosească aplicația întâmpină probleme la crearea unui cont.

Mai jos este prezentată funcția pentru afișarea utilizatorilor de tip doctor din cadrul sitemului.

```
public void showData()
{
    firebaseFirestore.collection( collectionPath: "doctor").get()
            .addOnCompleteListener(new OnCompleteListener<QuerySnapshot>() {
                @Override
                public void onComplete(@NonNull Task<QuerySnapshot> task) {
                    if (task.isSuccessful()) {
                        doctorModels.clear();
                        for (DocumentSnapshot snapshot : task.getResult()) {
                            DoctorModel model = new DoctorModel(snapshot.getString( field: "nume"), snapshot.getString( field: "prenume"),
                                    snapshot.getString( field: "telefon"),snapshot.getString( field: "email"),
                                    snapshot.getString( field: "specializare"));
                            doctorModels.add(model);
                        }
                        doctorAdapter.notifyDataSetChanged();
                    }
                }
            }).addOnFailureListener(new OnFailureListener() {
        @Override
        public void onFailure(@NonNull Exception e) {
            Toast.makeText(getActivity(), text: "Eroare la introducerea datelor",Toast.LENGTH_SHORT).show();
        }

    });

}
```

Fig 5. 12 Afișare doctori

5.5.5. Chat

Sistemul de chat este destinat comunicării dintre doctori si pacienți. Pentru a putea folosi chat-ul atât doctorul cât și pacientul trebuie să selecteze opțiunea de chat din meniul principal, iar apoi vor selecta doctorul sau pacientul cu care doresc să comunice din lista utilizatorilor aflați în sistem.

Mesajele trimise pe chat sunt stocate în Firebase, iar pentru afișarea lor a fost creat un câmp timestamp care reprezintă data și ora la care a fost creat mesajul. Acest câmp este util pentru a putea ordona mesajele în funcție de momentul în care au fost scrise.

Pentru a ne asigura că într-o conversație apar doar mesajele dintre doctorul si pacientul respectiv atunci când citim documentele din colecția de mesaje verificăm dacă câmpurile de *sender* și *reciever* din cadrul obiectelor de tip mesaj corespund cu ID-urile doctorului și pacientului. Se verifică atât dacă doctorul este cel care trimite mesajul și pacientul primește mesajul, cât și invers.

```
private void readMessage(String myid,String userId)
{
    mesajModels=new ArrayList<>();

    firebaseFirestore.collection( collectionPath: "messages").orderBy("timestamp")
            .addSnapshotListener(new EventListener<QuerySnapshot>() {
                @Override
                public void onEvent(@Nullable QuerySnapshot value, @Nullable FirebaseFirestoreException error) {
                    mesajModels.clear();
                    if (error != null) {
                        Log.d( tag: "Error:",error.getMessage());
                    } else {
                        for (QueryDocumentSnapshot querySnapshot : value) {
                            String sender = querySnapshot.getString( field: "sender");
                            String reciever = querySnapshot.getString( field: "reciever");
                            String message = querySnapshot.getString( field: "message");
                            if (sender.equals(myid) && reciever.equals(userId) || sender.equals(userId) && reciever.equals(myid)) {
                                MesajModel mesajModel = new MesajModel();
                                mesajModel.setSender(sender);
                                mesajModel.setReciever(reciever);
                                mesajModel.setMessage(message);
                                mesajModels.add(mesajModel);
                            }
                            messageAdapter = new MessageAdapter( context: Mesaj.this, mesajModels);
                            recyclerView.setAdapter(messageAdapter);
                        }

                    }
                }
            });

}
```

Fig 5. 13 Citire mesaj

5.5.6. Programări

Pentru programări pacientul selectează opțiunea de programare din meniul principal, apoi va trebui să selecteze doctorul la care dorește să își creeze o programare. Pe pagina de programare va putea vedea toate programările create de acesta, iar pentru a adăuga o programare nouă va trebui să selecteze data și ora la care dorește să își creeze programarea. Intervalul destinat unei programări este de 30 de minute, nu se poate adăuga o programare la aceeași dată ți oră sau în intervalul de jumătate de oră destinat unei anumite programări. Dacă se alege un interval orar în afara programului atunci se va afișa un mesaj de avertizare în partea de jos a ecranului și nu se va putea crea programarea.

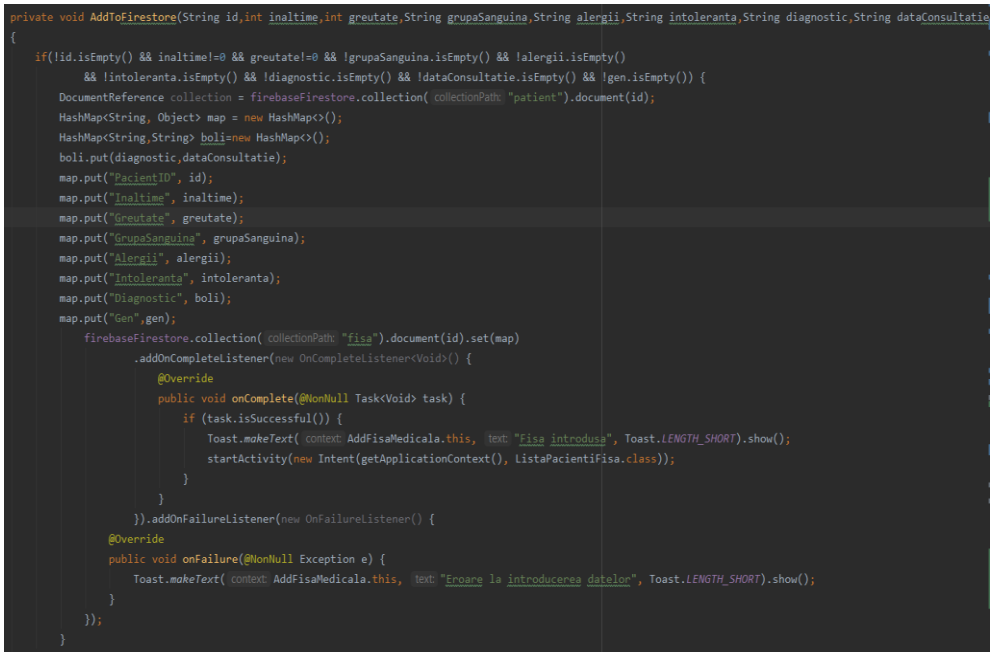
Doctorii care intră la programări vor vedea programările create pentru aceștia. Vor putea doar vizualiza programările, nu vor putea edita și șterge.

```
private void showDateTimeDialog(final EditText date_time_in) {
    final Calendar calendar=Calendar.getInstance();
    DatePickerDialog.OnDateSetListener dateSetListener=new DatePickerDialog.OnDateSetListener() {
        @Override
        public void onDateSet(DatePicker view, int year, int month, int dayOfMonth) {
            calendar.set(Calendar.YEAR,year);
            calendar.set(Calendar.MONTH,month);
            calendar.set(Calendar.DAY_OF_MONTH,dayOfMonth);
            TimePickerDialog.OnTimeSetListener timeSetListener=new TimePickerDialog.OnTimeSetListener() {
                @Override
                public void onTimeSet(TimePicker view, int hourOfDay, int minute) {
                    if(minute>=0 && minute <30)
                        minute=0;
                    else if (minute >=30 && minute<=59)
                        minute=30;
                    calendar.set(Calendar.HOUR_OF_DAY, hourOfDay);
                    calendar.set(Calendar.MINUTE, minute);

                    SimpleDateFormat simpleDateFormat=new SimpleDateFormat( pattern: "yy-MM-dd HH:mm");

                    if(hourOfDay>=16 || hourOfDay<8)
                    {
                        Toast.makeText( context: CalendarProgramare.this, text: "Interval depasit",Toast.LENGTH_SHORT).show();
                    }
                    else {
                        date_time_in.setText(simpleDateFormat.format(calendar.getTime()));
                    }
                }
            };
            new TimePickerDialog( context: CalendarProgramare.this,timeSetListener,calendar.get(Calendar.HOUR_OF_DAY)
                    ,calendar.get(Calendar.MINUTE), is24HourView: true).show();
        }
    };
    new DatePickerDialog( context: CalendarProgramare.this,dateSetListener,calendar.get(Calendar.YEAR),calendar.get(Calendar.MONTH),calendar.get(Calendar
```

Fig 5. 14 Creare programare

5.5.7. Fișă medicală

Fiecare pacient din cadrul aplicației are o fișă medicală proprie și unică care conține date despre starea de sănătate a acestuia. Aceasta este stocată în Firebase fiind accesibilă tot timpul, atât pacientului, cât și doctorului. Doctorul poate crea o fișă pentru un pacient sau poate edita o fișă deja existentă. Pacientul poate doar vizualiza fișa lui proprie.



```
private void AddToFirestore(String id,int inaltime,int greutate,String grupaSanguina,String alergii,String intoleranta,String diagnostic,String dataConsultatie
{
    if(!id.isEmpty() && inaltime!=0 && greutate!=0 && !grupaSanguina.isEmpty() && !alergii.isEmpty()
            && !intoleranta.isEmpty() && !diagnostic.isEmpty() && !dataConsultatie.isEmpty() && !gen.isEmpty()) {
        DocumentReference collection = firebaseFirestore.collection( collectionPath: "patient").document(id);
        HashMap<String, Object> map = new HashMap<>();
        HashMap<String,String> boli=new HashMap<>();
        boli.put(diagnostic,dataConsultatie);
        map.put("PacientID", id);
        map.put("Inaltime", inaltime);
        map.put("Greutate", greutate);
        map.put("GrupaSanguina", grupaSanguina);
        map.put("Alergii", alergii);
        map.put("Intoleranta", intoleranta);
        map.put("Diagnostic", boli);
        map.put("Gen",gen);
            firebaseFirestore.collection( collectionPath: "fisa").document(id).set(map)
                    .addOnCompleteListener(new OnCompleteListener<Void>() {
                        @Override
                        public void onComplete(@NonNull Task<Void> task) {
                            if (task.isSuccessful()) {
                                Toast.makeText( context: AddFisaMedicala.this, text: "Fisa introdusa", Toast.LENGTH_SHORT).show();
                                startActivity(new Intent(getApplicationContext(), ListaPacientiFisa.class));
                            }
                        }
                    }).addOnFailureListener(new OnFailureListener() {
                @Override
                public void onFailure(@NonNull Exception e) {
                    Toast.makeText( context: AddFisaMedicala.this, text: "Eroare la introducerea datelor", Toast.LENGTH_SHORT).show();
                }
            });
    }
```

Fig 5. 15 Adăugare fișă medicală

5.5.8. Prescripție medicală

Doctorii din cadrul aplicației pot crea prescripții pentru pacienți, în care oferă detalii despre medicamentul prescris cum ar fi numele acestuia, efectele secundare, administrarea precum și prețul. De asemenea pot edita prescripțiile create. Pacienții pot vedea prescripțiile care sunt create pentru aceștia.

```
public void showPrescriptie(String id)
{
    firebaseFirestore.collection( collectionPath: "prescriptie").get()
            .addOnCompleteListener(new OnCompleteListener<QuerySnapshot>() {
                @Override
                public void onComplete(@NonNull Task<QuerySnapshot> task) {
                    if (task.isSuccessful()) {
                        prescriptieModels.clear();
                        for (DocumentSnapshot snapshot : task.getResult()) {
                            PrescriptieModel model = new PrescriptieModel(snapshot.getString( field: "Nume"), snapshot.getString( field: "EfecteSecundare"),
                                    snapshot.getString( field: "Administrare"),snapshot.getLong( field: "Pret").intValue(),snapshot.getString( field: "PacientID"));

                            if(model.getPatientID().equals(id))
                                prescriptieModels.add(model);
                        }
                        prescriptieAdapter.notifyDataSetChanged();
                    }
                }
            }).addOnFailureListener(new OnFailureListener() {
        @Override
        public void onFailure(@NonNull Exception e) {
            Toast.makeText( context: VizualizarePrescriptieDoctor.this, text: "Eroare la afisarea datelor",Toast.LENGTH_SHORT).show();
        }

    });

}
```

Fig 5. 16 Vizualizare prescripții

Capitolul 6. Testare şi Validare

În acest capitol se prezintă cum au fost realizate cazurile de testare ale sistemului și rezultatele obținute în urma testării.

6.1. Cazuri de testare

În acest subcapitol se vor prezenta principalele cazuri de testare ale sistemului.

Cazurile de testare se referă la o anumită funcționalitate a sistemului care a fost aleasă pentru a fi testată și validată. Fiecare din aceste funcționalități trebuie testată după implementare pentru a verifica dacă există sau nu diferențe dintre rezultatul așteptat și cel afișat. Testarea și validarea a fost realizată după adăugarea unei componente noi în cadrul sistemului pentru a verifica dacă componenta respectivă funcționează așa cum își dorește dezvoltatorul aplicației, dar și dacă aceasta comunică fără probleme cu celelalte componente ale sistemului deja implementate.

6.1.1. Autentificare în sistem

Tabel 6. 1 Caz de testare-autentificare

	Paşi	Rezultat
1.	Deschidere aplicație	Aplicația va afișa pagina de atutentificare(MainActivity)
2.	Introducere credențiale autentificare(mail și parolă)	Textele introduse vor apărea în câmpurile de autentificare
3.	Validare date introduse	Utilizatorul va vedea un mesaj în partea de jos a ecranului în funcție de datele introduse(date valide/invalide)
4.	Redirecționare pagină utilizator	Utilizatorul este redirecționat în funcție de datele introduse

6.1.2. Înregistrare pacient în sistem

Tabel 6. 2 Caz de testare-înregistrare pacient

	Paşi	Rezultat
1.	Deschidere aplicație	Aplicația va afișa pagina de atutentificare(MainActivity)
2.	Selectare opțiune de Creare cont	Aplicația va deschide o nouă fereastră pentru crearea unui cont nou
3.	Introducere date specificate în câmpuri	Textele introduse vor apărea în câmpurile specificate

4.	Validare date introduse	Utilizatorul va vedea un mesaj în partea de jos a ecranului în funcție de datele introduse(date valide/invalide)
5.	Autentificare cu noul cont	Utilizatorul se poate autentifica folosind contul creat

6.1.3. Trimitere mesaj

Tabel 6. 3 Caz de testare-trimitere mesaj

	Pași	Rezultat
1.	Deschidere aplicație ca și pacient	Aplicația va afișa pagina pacientului(PatientInterface)
2.	Selectare opțiune de Chat	Aplicația va deschide o nouă fereastră pentru crearea unui cont nou pentru un doctor
3.	Selectare doctor	Aplicația va deschide o nouă fereastră unde se va putea vedea conversația cu doctorul selectat
4.	Compunere mesaj	Pacientul va completa câmpul specific mesajului
5.	Trimitere mesaj	Pacientul va trimite mesajul către doctor
6.	Vizualizare mesaj	Pacientul va putea vedea în fereastra de mesaje noul mesaj

6.1.4. Adăugare doctor

Tabel 6. 4 Caz de testare-adăugare doctor

	Pași	Rezultat
1.	Deschidere aplicație ca și administrator	Aplicația va afișa pagina administratorului(Admin)
2.	Selectare opțiune de Adaugare	Aplicația va deschide o nouă fereastră pentru crearea unui cont nou pentru un doctor
3.	Introducere date specificate în câmpuri	Textele introduse vor apărea în câmpurile specificate
4.	Validare date introduse	Administratorul va vedea un mesaj în partea de jos a

		ecranului în funcție de datele introduse(date valide/invalide)
5.	Vizualizare cont nou	Administratorul va putea vedea contul noului doctor creat în lista celorlalți doctori din sistem

6.1.5. Creare programare

Tabel 6. 5 Caz de testare-creare programare

	Pași	Rezultat
1.	Deschidere aplicație ca și pacient	Aplicația va afișa pagina pacientului(PatientInterface)
2.	Selectare opțiune de Programare online	Aplicația va deschide o nouă fereastră pentru selectarea doctorului
3.	Selectare doctor	Aplicația va deschide o nouă fereastră pentru crearea programării
4.	Introducere date specificate câmpuri	Textele introduse vor apărea în câmpurile specificate
5.	Validare date introduse	Pacientul va vedea un mesaj în partea de jos a ecranului în funcție de datele introduse(date valide/invalide)
6.	Vizualizare programări create	Pacientul va putea vizualiza toate programările create de acesta

Capitolul 7. Manual de Instalare si Utilizare

În acest capitol se prezintă resursele hardware și software necesare pentru a instala și rula aplicația, manualul de instalare și manualul de utilizare al aplicației cu detaliile necesare pentru ca un utilizator să înțeleagă cum se folosește aplicația.

7.1. Manual de instalare

Pentru componenta Android a aplicației sunt necesare următoarele resurse pentru ca aceasta să poată fi folosită de către un dispozitiv mobil:

- Smartphone cu sistem de operare Android
- Spațiu minim de memorie disponibil 10MB
- Versiune Android minimă 4.0
- Conexiune la internet(Wi-fi,3G,4G)

Resurse software necesare rulării programului pe un calculator pentru instalarea aplicației:

- Sistem de operare Windows bazat pe arhitectura x64
- Procesor Frecventa minima 2.0GHz,4 nuclee
- Memorie RAM: 4GB
- IDE:Android Studio 4.1.2
- Java SDK 1.8

7.2. Manual de utilizare

Se vor prezenta pașii care trebuie urmați pentru ca orice utilizator, în funcție de tipul acestuia sa poată înțlegele cât mai ușor cum se poate utiliza această aplicație.

7.2.1. Autentificare

Aplicația se va deschide la fereastra de autentificare. Aici fiecare utilizator în funcție de tipul acestuia după introducerea adresei de mail ăi parolei ăi după validarea acestora va fi redirecționat către pagina sa proprie. Dacă adresa de mail sau parola nu există în baza de date a sistemului atunci se va afișa un mesaj de eroare, iar utilizatorul nu se va putea conecta și redirecționa către pagina acestuia. De asemenea este nevoie și de conexiune la internet pentru ca utilizatorul să se poată autentifica.

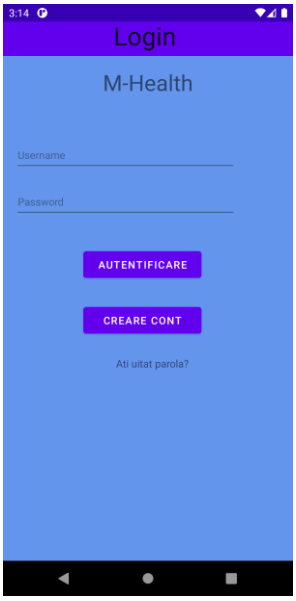


Fig 7. 1 Pagina de autentificare

7.2.2. Creare cont

Tot pe pagina de autentificare pacienții care doresc să folosească aplicația, dar nu dispun de un cont pe această platformă au posibilitatea de a-și crea un cont selectând opțiunea de creare cont. Acolo vor trebui să completeze câteva date personale și anume nume, prenume, data nașterii, adresa de mail, telefon și parolă. După ce au introdus aceste date, se vor putea autentifica în aplicație folosind acea adresă de mail și parolă. În cazul celorlalți utilizatori și anume doctorul și serviciul de ambulanță aceștia vor avea un cont în sistem creat de către administrator.

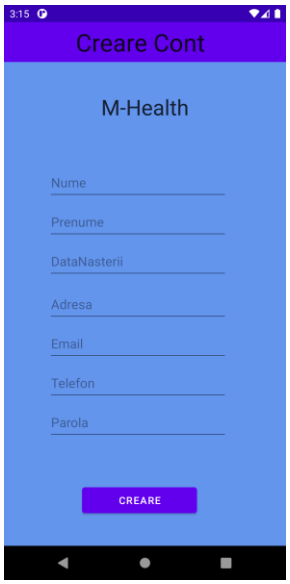


Fig 7. 2 Pagina de creare cont

7.2.3. Resetare parolă

Utilizatorul va trebui să introducă adresa de mail, iar pe acea adresă va primi un mail cu un link de unde va putea să își introducă noua parolă. După ce și-a introdus noua parolă se va putea autentifica în sistem cu acea parolă.

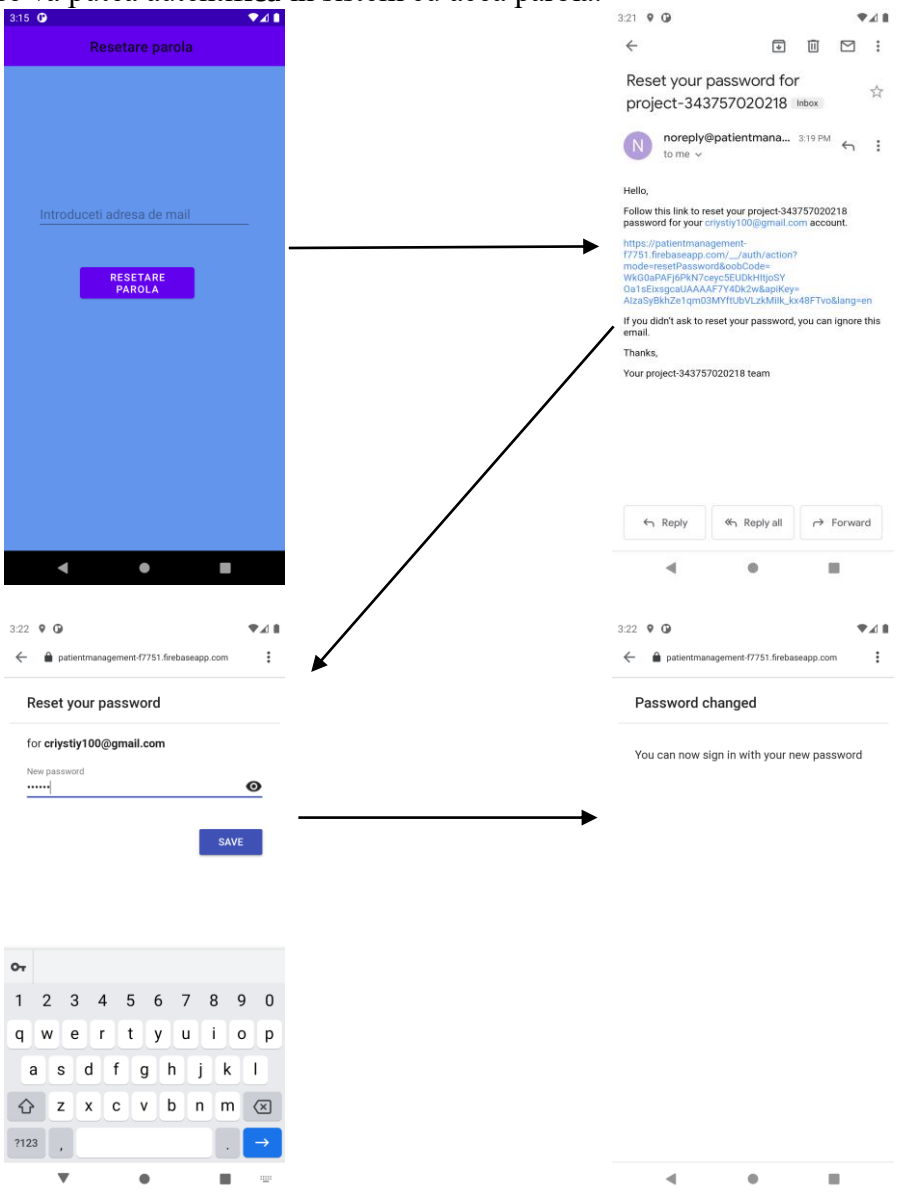


Fig 7. 3 Paginile pentru resetarea parolei

7.2.4. Administratorul

După autentificare este redirecționat pe pagina acestuia unde sunt trei ferestre dedicate fiecărui tip de utilizator și anume doctor, pacient și serviciu de ambulanță. Administratorul va vedea datele despre fiecare din acei utilizatori. El va putea adăuga noi utilizatori folosind butonul de „Adauga”, va putea edita datele utilizatorilor folosind un swipe left, se va deschide o nouă fereastră cu datele utilizatorilor cere vor putea fi modificate, iar prin apăsarea butonului de „Actualizare” datele vor fi actualizate în baza

de date a sitemului. Dacă administratorul dorește să șteargă un anumit utilizator atunci va folosi opțiunea de swipe right și automat utilizatorul respectiv va fi șters din sistem.

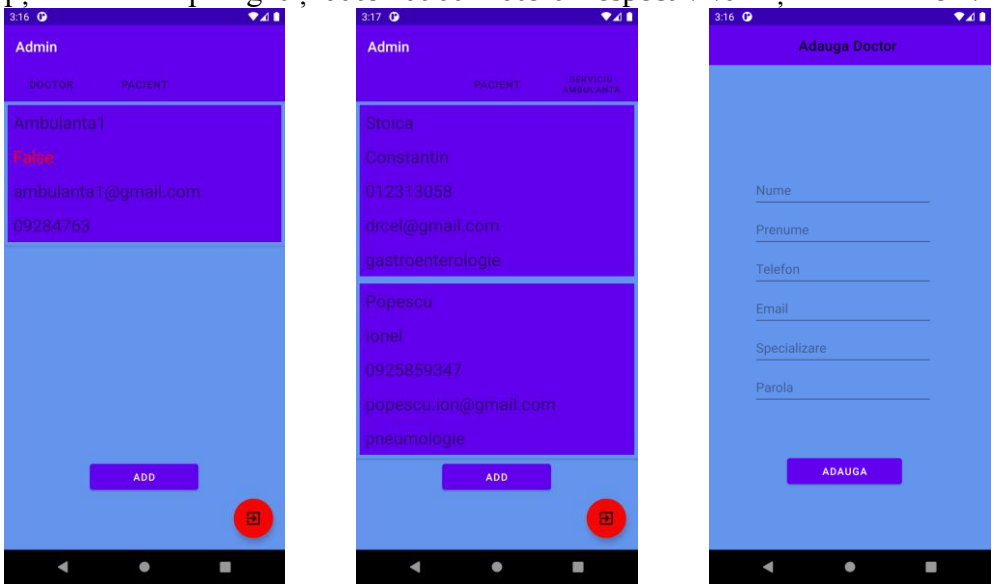
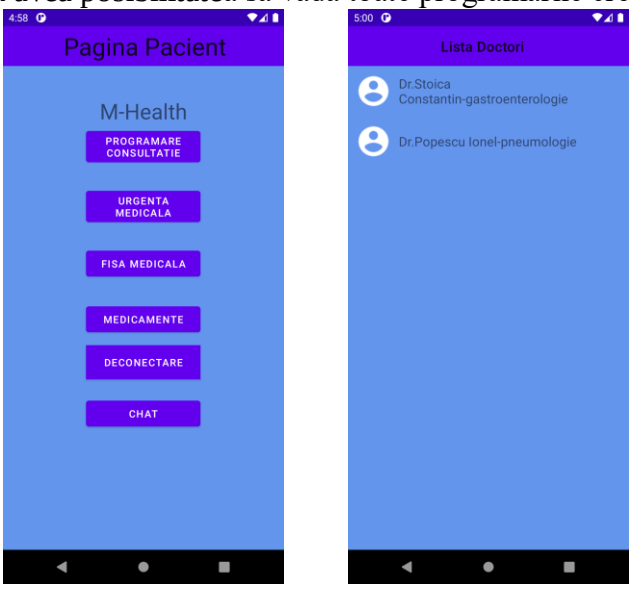


Fig 7. 4 Intefață administrator

7.2.5. Programări

Pacientul poate să își creeze programare la oricare doctor existent în sistem. Acesta va selecta opțiunea de programări din meniul principal, apoi va selecta din lista de doctori existenți în sistem pe acela la care dorește să își creeze o programare. Va selecta data și ora la care dorește să își creeze programarea. Se va afișa un mesaj de succes dacă programarea a fost adăugată. De asemenea va putea vizualiza programările existente la acel doctor. Pacientul va avea posibilitatea să vadă toate programările create de acesta.



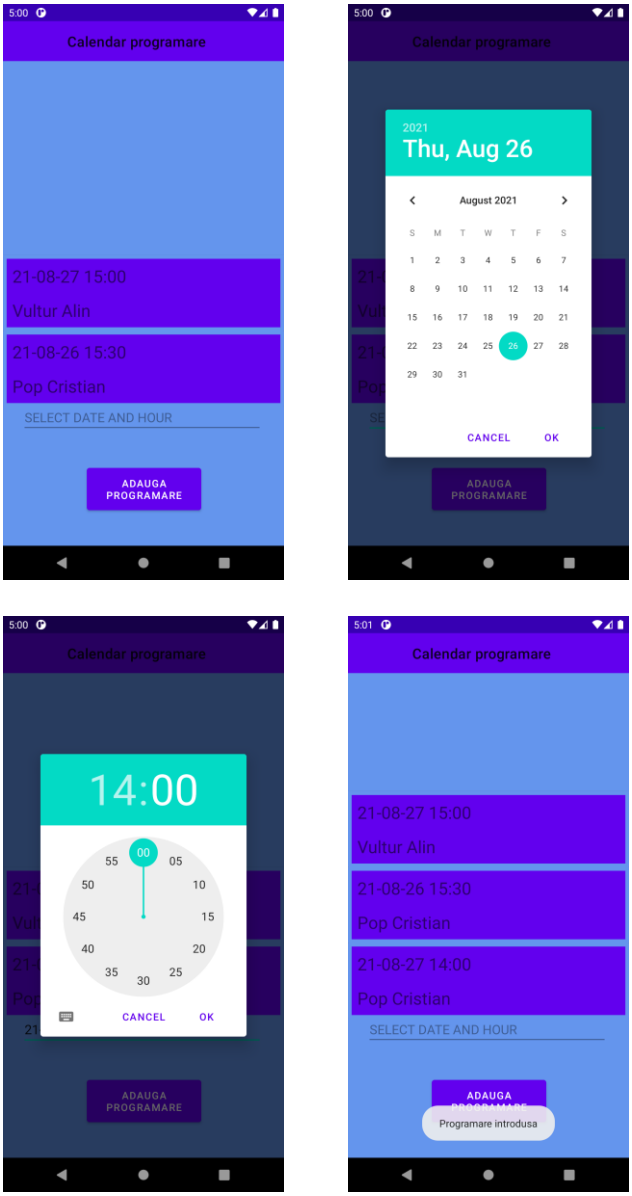


Fig 7. 5 Adăugare programare

Doctorul poate selecta opțiunea de vizualizare programări din meniul acestuia și vor fi afișate toate programările create pentru acesta.

Fig 7. 6 Vizualizare programări doctor

7.2.6. Fișă medicală

Doctorul poate adăuga sau edita fișa medicală a unui pacient de pe pagina destinată fișei medicale. El va selecta pacientul și va putea adăuga o fișă medicală. Dacă pacientul are deja o fișă medicală în sistem atunci doctorul va putea edita acea fișă. Pacientul va putea vizualiza propria lui fișă medicală selectând opțiunea de fișă medicală din meniul principal. Acesta va putea doar să o vizualizeze, nu va putea edita fișa, operația fiind valabilă doar pentru doctor.

Fig 7. 7 Vizualizare fișa medicală

7.2.7. Chat

Sistemul de chat este destinat pentru comunicarea dintre doctori și pacienți. Se pot trimite mesaje între aceștia prin scrierea mesajului în spațiul destinat acestuia în partea de jos a ecranului, iar trimiterea lui se realizează folosind butonul de „Send” din dreapta jos.

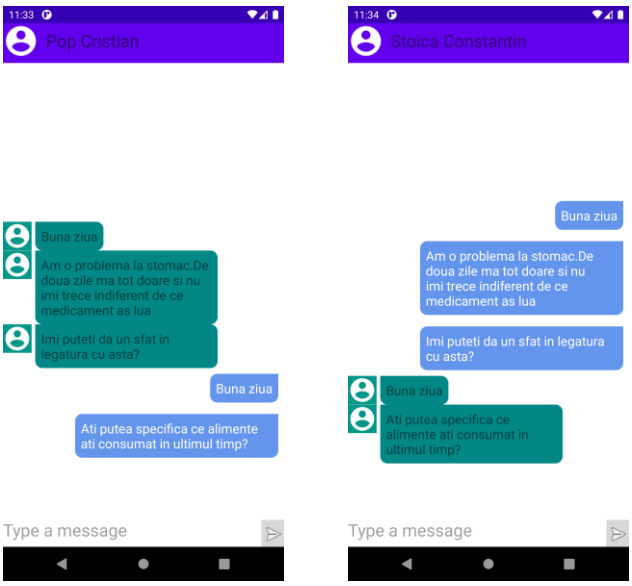
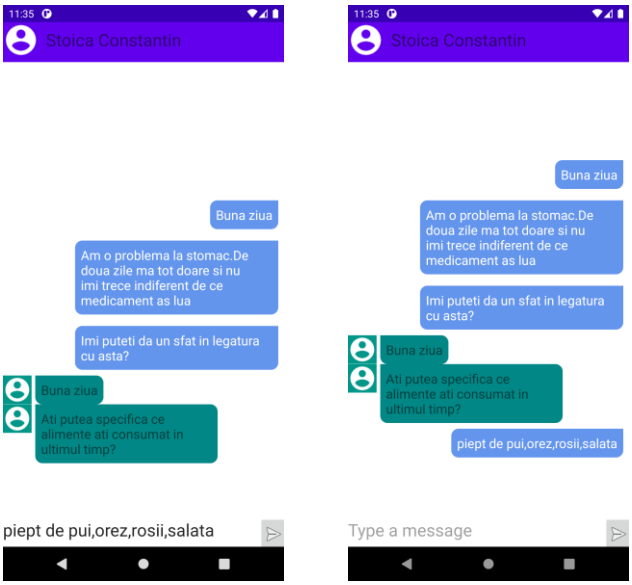


Fig 7.x Sistemul de chat



7.2.8. Prescripție medicală

Doctorul poate crea prescripții pentru pacient de pe pagina destinată prescripției. El va selecta pacientul, apoi va vizualiza toate prescripțiile existente pentru pacientul respectiv. Va putea edita o prescripție deja existentă sau va putea adăuga o prescripție nouă folosind butonul „Adauga prescriptie”. Pacientul va putea vizualiza prescripțiile create pentru acesta.

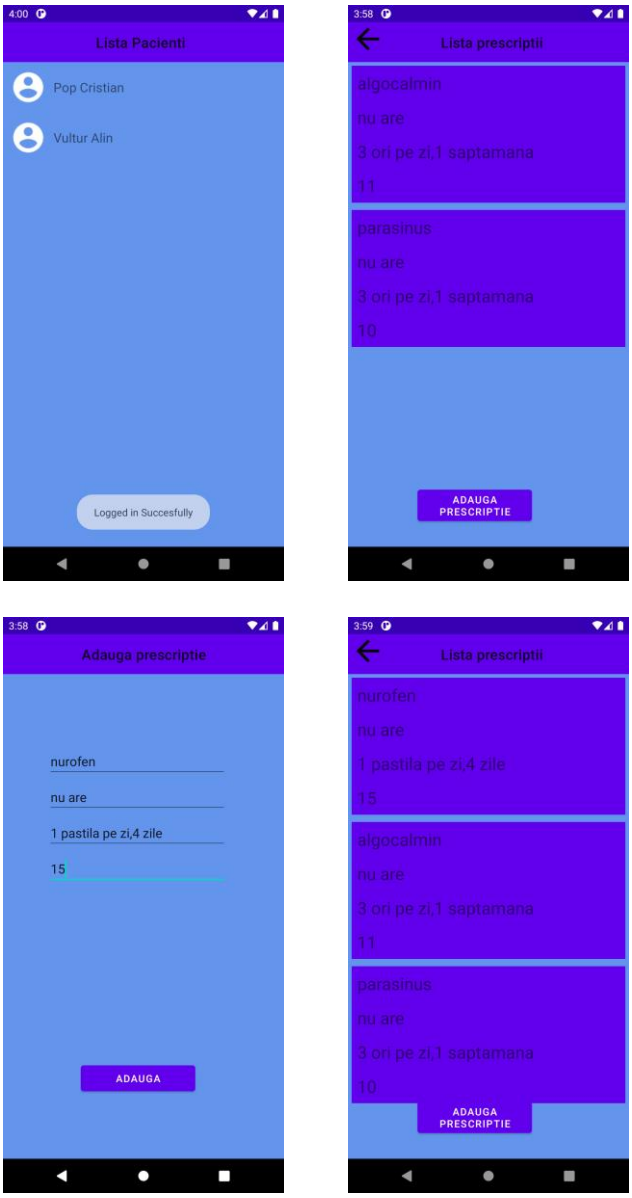


Fig 7. 8 Creare prescripție

Capitolul 8. Concluzii

În acest capitol se prezintă un rezumat al aplicației, rezultatele obținute în urma implementării, dar și dezvoltări ulterioare pentru a îmbunătăți și mai mult interacțiunea utilizatorului cu aplicația.

8.1. Rezultate obținute

Obiectivul proiectului a fost crearea unei aplicații Android care să fie destinată unei clinici medicale și care să poată fi utilizată atât de pacienți, cât și de doctori, serviciu de ambulanță și administrator. Se dorește îmbunătățirea comunicării dintre pacient și doctor, stocarea informațiilor dintr-o fișă medicală pentru a fi la îndemâna pacienților și a doctorilor în orice moment, crearea unui sistem de programări. De asemenea se dorește gestionarea utilizatorilor de către administrator într-un mod cât mai simplu, gestionarea urgențelor medicale prin reducerea timpului necesar transmiterii informațiilor de la pacient la serviciul de ambulanță.

Îmbunătățirea comunicării dintre pacient și doctor a fost realizată prin implementarea unui sistem de chat în cadrul aplicației prin care pacienții au posibilitatea de a comunica direct cu orice doctor din cadrul sistemului.

Crearea unui sistem de programări online, pacientul are posibilitatea de a-și crea o programare direct din aplicație la oricare doctor și de a vedea toate programările create de acesta. De asemenea doctorul poate vizualiza toate programările care au fost create pentru acesta.

Digitalizarea fișei medicale este un alt obiectiv propus și realizat de acest sistem. Astfel aplicația conține fișa medicală a pacientului, fiind disponibilă pentru acesta în orice moment. Doctorul poate vizualiza fișa medicală a pacienților din cadrul aplicației, astfel este îmbunătățit timpul de căutare al acesteia. Mai mult și prescripțiile medicale sunt în format digital, acestea fiind ușor de accesat pentru pacienți în momentul în care au nevoie de ele.

Datele utilizatorilor sunt mai ușor de gestionat datorită interfeței administratorului, care este foarte ușor de folosit. Poate adăuga, șterge și edita datele acestora.

8.2. Dezvoltări ulterioare

Aplicația prezentată în această lucrare are posibilitatea de extindere astfel încât să devină o aplicație și mai performantă, oferind și mai mult sprijin pacienților care doresc să folosească această aplicație, cât și doctorilor pentru o gestionare cât mai ușoară a pacienților. Printre posibilele dezvoltări ulterioare se numără:

- Posibilitatea înregistrării unui cont în aplicație folosind alte metode decât adesa de mail cum ar fi contul de Facebook, numărul de telefon, contul de Google.
- Implementarea unui apel video pentru consultația medicală. Consultația medicală ar putea fi realizată și de la distanță printr-un apel video, oferind posibilitatea ca pacientul și doctorul să poată comunica mult mai ușor unul cu celălalt

- Implementarea unui sistem de plată, utilizatorul având posibilitatea de a plăti diverse servicii medicale direct din aplicație
- Un sistem de notificări pentru programare. Utilizatorul să fie înștiințat printr-o notificare în ziua programării. De asemenea notificările să poată fi primite și în cazul sistemului de chat când un utilizator, doctor sau pacient, primește un mesaj de la alt utilizator.
- Un isoric medical pentru pacient în care să existe consultațiile acestuia, rezultatele analizalor medicale, urgențele medicale pe care le-a avut, iar doctorii să poată vizualiza acest isotric
- Implementarea unei hărți care să arate drumul cel mai scurt de la locația curentă la locația clinicii medicale.

Bibliografie

[1] G Eysenbach, "What is E-Health?", *J Med Internet Res*., 2001, de pe site-ul NCBI – National Center for Biotechnology Information, disponibil online: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1761894/

[2] Lian Soon Lee "Emergency Call Mobile Application",Universiti Tunku Adbul Rahman, 2016, disponibil online: http://eprints.utar.edu.my/1956/1/IA-2016-1205323.pdf

[3] S. R. Steinhubl, E. D. Muse și E. J. Topol, „The emerging field of mobile health," *Sci Transl Med*, 2015, disponibil online: https://europepmc.org/article/med/25877894

[4] H .Thimbebly "Tehnology and the future of healthcare", College of Science, Swansea University,UK, *Journal of Public Health Research 2013*; volume 2:e28, disponibil online: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4147743/

[5] Junaid Bajwa, Usman Munir, Aditya Nori, Bryan Williams "Artificial Intelligence in healthcare:transforming the practice of medicine*", Future Healthcare Journal 2021 Volume 8*,No 2:e188-94, disponibil online: https://www.rcpjournals.org/content/futurehosp/8/2/e188.full.pdf

[6] H. C. Osenbaard, L. V. G-Pijnen "e-Health and quality in health care implementation time",*International Journal of Quality in Health Care*, 2016, 28(3), 415-419, disponibil online https://academic.oup.com/intqhc/article/28/3/415/1750408

[7] C. L. Ventola, "Mobile Devices and Apps for Health Care Professionals:Uses and Benefits", P T. 2014 May; 39(5): 356-364, disponibil online https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4029126/

[8] J. O'Donovan, A. Bersin, C. O'Donovan, "The effectiveness of mobile health(mhealth) tehnologies to train professionals in developing countries: a review of the literature", *BMJ Innovations 2015*; 1:33-36, disponibil online: https://innovations.bmj.com/content/1/1/33

[9] N. Smyth, "Android Studio 4.0 Development Essentials – Java Edition", PayloadMedia, 2020, disponibil online: https://www.techotopia.com/pdf_previews/AndroidStudio40EssentialsPreview.pdf

Anexa 1

Anexa 1 cuprinde lista de figuri, lista de tabele, Glosar de termeni

Lista de figuri

Lista tabelelor

Glosar de termeni

Termen	**Descriere**
mCLINIChelp	Denumirea aplicației
SMS (Short Message Service)	Tehnologie pentru trimiterea de mesaje între dispozitivele mobile
OMS (Organizația Mondială a Sănătății)	Organizație internațională care are rolul de a menține si coordona situația sănătății populațiilor de pe glob
CRUD (create,read,update,delete)	Acronim folosit pentru operațiile de bază creare,citire,actualizare,stergere date
GPS Global Positioning System)	Sistem global de navigație prin satelit și unde radio
JSON (JavaScript Object Notation)	Format de reprezentare de date
API (Application Programming Interface)	Tip de interfață software care oferă un serviciu altor componente software
SDK (Software Development Kit)	Set de unelte folosite pentru a scrie programe
IDE (Integrated Development Environment)	Mediu de dezvoltare al aplicațiilor software